INNOVATION, ENTREPRENEURSHIP AND MANAGEMENT SERIES

Machine Learning for Asset Management

New Developments and Financial Applications

Edited by
Emmanuel Jurczenko

ISTE WILEY

Machine Learning for Asset Management

Machine Learning for Asset Management

New Developments and Financial Applications

Edited by

Emmanuel Jurczenko

ISTE

WILEY

First published 2020 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc.

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

Library of Congress Control Number: 2019955407

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISBN 978-1-78630-544-2

Contents

Foreword

The development of machine learning offers enormous opportunities to our industry, but it also presents some challenges: as machines continue to evolve, how much control should investment managers relinquish to technology to deliver the best outcomes for investors?

At Unigestion, we believe in collaborative intelligence and in the strength of humans and machines working together. We empower our investment teams with leading-edge technology to gain a deeper understanding of financial markets, in order to get better investment outcomes for our clients.

We strongly believe that machine learning can help active managers differentiate themselves from passive ones. There is huge potential for asset managers to use machine learning to support their investment decision-making, especially if backed up by human experience. Thanks to their ability to process much more complex patterns with better forecasting power, modern machine learning algorithms outperform traditional linear regression.

Machine learning is very good at finding statistical patterns through a mass of numbers, but those patterns are merely correlations amongst vast reams of data, rather than causative truths. As with any data-driven method, the data quality has a huge impact on the usefulness of the model output.

The principle of ‘garbage in, garbage out’ is also valid in this new quantitative world. For this reason, we believe investment managers must give an economic meaning to machine learning algorithms.

This groundbreaking book on *Machine Learning for Asset Management* represents a refreshing collaborative effort between sophisticated investment practitioners and researchers, to present practical application of machine learning methodologies. As one can see from the different chapters, machine learning can be applied to different parts of the investment process, from stock picking to tactical allocation, alpha signal enhancement or trading.

As a result, this comprehensive volume is a powerful tool to help practitioners keep abreast of developments in this fast-changing field, and to implement machine-learning methods into their investment value chain.

The future of asset management will likely involve a synthesis of human and artificial intelligence that harnesses the power of both. However, we need to clearly define the sharing of control between machine and manager. Whereas computers excel in responding to well-formulated questions with clear objectives, humans remain key in asking the right questions and interpreting the results.

Fiona FRICK
Group CEO, Unigestion

Acknowledgments

Firstly, I would like to thank the editorial team from ISTE. I would also like to express my gratitude to all the authors of the book for their contribution and trust. Finally, I would like to give special and warm thanks to Jérôme Teiletche, Head of Cross-Asset solutions at Unigestion, who introduced me to the subtleties of machine learning techniques in asset management.

1

Time-series and Cross-sectional Stock Return Forecasting: New Machine Learning Methods

This chapter extends the machine learning methods developed in Han *et al.* (2019) for forecasting cross-sectional stock returns to a time-series context. The methods use the elastic net to refine the simple combination return forecast from Rapach *et al.* (2010). In a time-series application focused on forecasting the US market excess return using a large number of potential predictors, we find that the elastic net refinement substantively improves the simple combination forecast, thereby providing one of the best market excess return forecasts to date. We also discuss the cross-sectional return forecasts developed in Han *et al.* (2019), highlighting how machine learning methods can be used to improve combination forecasts in both the time-series and cross-sectional dimensions. Overall, because many important questions in finance are related to time-series or cross-sectional return forecasts, the machine learning methods discussed in this chapter should provide valuable tools to researchers and practitioners alike.

1.1. Introduction

Researchers in finance increasingly rely on machine learning techniques to analyze Big Data. The initial application of the *least absolute shrinkage and selection operator* (Tibshirani 1996, LASSO) – one of the most popular machine learning techniques – in finance appears to be Rapach *et al.* (2013), who analyze lead-lag relationships among monthly international equity

Chapter written by David E. RAPACH and Guofu ZHOU.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

returns in a high-dimensional setting. More recently, Gu *et al*. (2019) employ a comprehensive set of machine learning tools, including the LASSO, to analyze the time-series predictability of monthly individual stock returns, while Chinco *et al*. (2019) use the LASSO to predict individual stock returns one minute ahead. Freyberger *et al*. (2019) apply a nonparametric version of the LASSO to accommodate nonlinear relationships between numerous firm characteristics and cross-sectional stock returns. Kozak *et al*. (2019) use the LASSO in a Bayesian context to model the stochastic discount factor based on a large number of firm characteristics. Incorporating insights from Bates and Granger (1969); Diebold and Shin (2019), Han *et al*. (2019) propose procedures for forecasting cross-sectional returns using the information in more than 100 firm characteristics[1].

In this chapter, we show how the approach of Han *et al*. (2019), originally designed for forecasting cross-sectional stock returns, can be modified for time-series forecasting of the market excess return. A voluminous literature investigates market excess return predictability based on a wide variety of predictor variables[2]. In the presence of a large number of potential predictor variables, conventional forecasting methods are susceptible to in-sample overfitting, which often translates into poor out-of-sample performance. Rapach *et al*. (2010) employ forecast combination (Bates and Granger 1969) to incorporate the information in a large number of predictor variables in a manner that guards against overfitting. They find that a simple combination forecast – the average of univariate predictive regression forecasts based on the individual predictor variables – substantially improves out-of-sample forecasts of the US market excess return. Extending the methods of Han *et al*. (2019) to a time-series context along the lines of Diebold and Shin (2019), we describe how the *elastic net* (Zou and Hastie 2005), a well-known variant of the LASSO, can be used to refine the simple combination forecast, resulting in what we call the *combination elastic net* forecast. Intuitively, as explained by Han *et al*. (2019), the elastic net refinement allows us to more efficiently use the information in the predictor variables by selecting the most relevant predictors to include in the combination forecast. In an empirical application, we show that the combination elastic net approach indeed improves the

1 We focus on numerical data in this chapter. For textual analysis see, for example, Tetlock (2007); Loughran and McDonald (2011); Ke *et al*. (2019).

2 See Rapach and Zhou (2013) for a survey of the literature.

accuracy of US market excess return forecasts and provides substantive economic value to a mean-variance investor. Overall, our combination elastic net forecast appears to be among the best market excess return forecasts to date.

The rest of the chapter is organized as follows. Section 1.2 describes the construction of market excess return forecasts, including the combination elastic net forecast. Section 1.3 reports results for an empirical application centered on forecasting the US market excess return, using a variety of predictor variables from the literature. Section 1.4 outlines the construction of the cross-sectional return forecasts proposed by Han *et al*. (2019). Section 1.5 concludes this chapter.

1.2. Time-series return forecasts

1.2.1. *Predictive regression*

Stock market excess return predictability is typically analyzed in the context of a univariate predictive regression model:

$$r_t = \alpha + \beta x_{j,t-1} + \varepsilon_t, \qquad [1.1]$$

where r_t is the period-t return on a broad stock market index in excess of the risk-free return, $x_{j,t}$ is the predictor variable, and ε_t is a zero-mean disturbance term. It is straightforward to use equation [1.1] to generate an out-of-sample forecast of r_{t+1} based on $x_{j,t}$ and data available through period t:

$$\hat{r}^{(j)}_{t+1|t} = \hat{\alpha}^{(j)}_{1:t} + \hat{\beta}^{(j)}_{1:t} x_{j,t}, \qquad [1.2]$$

where $\hat{\alpha}^{(j)}_{1:t}$ and $\hat{\beta}^{(j)}_{1:t}$ are the ordinary least squares (OLS) estimates of α and β, respectively, in equation [1.1] based on data available from the start of the sample through t (i.e. the period of forecast formation).

Because there are a plethora of plausible predictor variables, it is advisable to aggregate information when forecasting the market excess return. The most

obvious approach for incorporating information from multiple predictor variables is to specify a multiple predictive regression model:

$$r_t = \alpha + \sum_{j=1}^{J} \beta_j x_{j,t-1} + \varepsilon_t. \quad [1.3]$$

It is again straightforward to use equation [1.3] to generate an out-of-sample forecast of r_{t+1} based on $x_{j,t}$ for $j = 1, \ldots, J$ and data available through t:

$$\hat{r}^{\text{OLS}}_{t+1|t} = \hat{\alpha}^{\text{OLS}}_{1:t} + \sum_{j=1}^{J} \hat{\beta}^{\text{OLS}}_{j,1:t} x_{j,t}, \quad [1.4]$$

where $\hat{\alpha}^{\text{OLS}}_{1:t}$ and $\hat{\beta}^{\text{OLS}}_{j,1:t}$ are the OLS estimates of α and β_j, respectively, for $j = 1, \ldots, J$ in equation [1.3] based on data available through t.

Although the out-of-sample market excess return forecasts in equation [1.2] and [1.4] are easy to obtain, Goyal and Welch (2008) find that such forecasts, based on numerous popular predictor variables from the literature, fail to outperform the naive prevailing mean benchmark forecast on a consistent basis over time (as judged by the out-of-sample R^2 statistic, which we define below). The prevailing mean forecast ignores information in any predictor variable; it is simply the historical average excess return based on data available through t:

$$\hat{r}^{\text{PM}}_{t+1|t} = \frac{1}{t} \sum_{s=1}^{t} r_s. \quad [1.5]$$

The prevailing mean forecast corresponds to the following simple data-generating process for the market excess return:

$$r_t = \alpha + \varepsilon_t, \quad [1.6]$$

namely, the constant expected excess return (or random walk with drift) model. The Goyal and Welch (2008) findings pose important challenges for out-of-sample return predictability, as they indicate that exploiting the information in popular predictor variables via conventional regression methods does not improve forecast accuracy.

1.2.2. *Forecast combination*

The study of Goyal and Welch (2008) was influential in stimulating thinking about how to better use the information in predictor variables to forecast the market excess return. With respect to the univariate predictive regression forecast in equation [1.2], Rapach *et al*. (2010) argue that it is risky to rely on a single predictor variable, due to factors such as investor learning and structural change. Building on the seminal work of Bates and Granger (1969), Rapach *et al*. (2010) recommend forecast combination as a strategy for incorporating information from a variety of predictor variables. Forecast combination reduces forecast "risk" by diversifying across individual forecasts, similarly to diversifying across assets to reduce portfolio risk (Timmermann 2006). Specifically, Rapach *et al*. (2010) consider a combination forecast that takes the form of a simple average of the univariate predictive regression forecasts, based on $x_{j,t}$ for $j = 1, \ldots, J$ in equation [1.2]:

$$\hat{r}^{\mathrm{C}}_{t+1|t} = \frac{1}{J} \sum_{j=1}^{J} \hat{r}^{(j)}_{t+1|t}. \qquad [1.7]$$

They show that, in contrast to the conventional univariate and multiple predictive regression forecasts in equations [1.2] and [1.4], respectively, the combination forecast in equation [1.7] is able to deliver out-of-sample accuracy gains relative to the prevailing mean forecast, on a much more consistent basis over time.

How is it that – unlike the conventional multiple predictive regression forecast in equation [1.4], which also includes information from $x_{j,t}$ for $j = 1, \ldots, J$ – the combination forecast in equation [1.7] is able to improve out-of-sample performance? Rapach *et al*. (2010) point out that forecast combination is effectively a strong shrinkage estimator. They show that the combination forecast in equation [1.7] makes two adjustments to the conventional multiple predictive regression forecast in equation [1.4]: first, it replaces the OLS multiple regression coefficient estimates with their univariate counterparts, which reduces the role of multi-collinearity in producing imprecise parameter estimates; second, the combination forecast shrinks the univariate slope coefficients by the factor $1/J$, thereby shrinking the forecast to the prevailing mean benchmark.

The usefulness of shrinkage for improving out-of-sample market excess return forecasts stems from a delicate balance required for stock return forecasting. On the one hand, we want to incorporate information from a wide variety of potentially relevant predictor variables, especially since we do not want to neglect relevant information and cannot know *a priori* which predictors are the most relevant. On the other hand, incorporating information from numerous predictors via the multiple prediction regression forecast in equation [1.4] is inadvisable. Equation [1.4] is based on conventional estimation of the multiple predictive regression model in equation [1.3], which is susceptible to overfitting. Conventional OLS estimation maximizes the explanatory ability of the model over the estimation sample, which often leads to poor out-of-sample performance. Overfitting concerns are exacerbated as the number of explanatory variables increases and the signal-to-noise ratio in the data decreases. We encounter both of these challenges when forecasting stock returns: there are numerous plausible predictor variables, and the predictable component in returns is inherently limited. The combination forecast in equation [1.7] apparently provides an effective shrinkage strategy for incorporating information from numerous plausible predictor variables in a manner that avoids overfitting.

1.2.3. *Elastic net*

Machine learning techniques also provide a means for implementing shrinkage. Indeed, the popular LASSO estimator is a penalized regression approach that is explicitly designed to prevent overfitting via shrinkage. To compute a forecast based on the multiple predictive regression model in equation [1.3], instead of the OLS objective function, we estimate the coefficients using the LASSO objective function:

$$\underset{\alpha,\beta_1,\ldots,\beta_J\in\mathbb{R}}{\arg\min}\left[\frac{1}{2t}\sum_{s=1}^{t}\left(r_s-\alpha-\sum_{j=1}^{J}\beta_j x_{j,s-1}\right)^2+\lambda\sum_{j=1}^{J}|\beta_j|\right], \quad [1.8]$$

where $\lambda \geq 0$ is a regularization parameter that controls the degree of shrinkage[3]. The first component of the LASSO objective function is the

3 Following standard practice, the predictor variables are standardized to have zero mean and unit variance before entering equation [1.8]. The final parameter estimates reflect the original scales of the predictor variables.

familiar sum of squared fitted residuals, so that the LASSO and OLS estimators coincide when $\lambda = 0$. The regularization parameter λ shrinks the coefficients towards zero. Unlike rigde regression (Hoerl and Kennard 1970), which relies on an ℓ_2 penalty term, the LASSO employs an ℓ_1 penalty, so that it permits shrinkage to exactly zero (for sufficiently large λ). Shrinkage to zero means that the LASSO also performs variable selection, which facilitates the interpretation of the fitted model.

To implement LASSO estimation, it is necessary to choose the value for λ. The most popular approach is K-fold cross-validation. However, the selection of the number of folds K and construction of the folds are largely arbitrary. The Hurvich and Tsai (1989) corrected version of the Akaike information criterion (Akaike 1973, AIC) provides an alternative to K-fold cross-validation for choosing λ. The corrected AIC is simpler to use in that it does not require arbitrary choices for the number and type of folds. Furthermore, Flynn *et al.* (2013) show that the corrected AIC has good asymptotic and finite-sample properties for choosing λ.

Zou and Hastie (2005) propose the elastic net (ENet) as a refinement to the LASSO that includes both ℓ_1 and ℓ_2 components in the penalty term. The ENet estimator is defined by the following objective function:

$$\underset{\alpha,\beta_1,\ldots,\beta_J\in\mathbb{R}}{\arg\min} \left[\frac{1}{2t} \sum_{s=1}^{t} \left(r_s - \alpha - \sum_{j=1}^{J} \beta_j x_{j,s-1} \right)^2 + \lambda P_\delta \left(\beta_1, \ldots, \beta_J \right) \right], \quad [1.9]$$

where

$$P_\delta \left(\beta_1, \ldots, \beta_J \right) = 0.5(1-\delta) \sum_{j=1}^{J} \beta_j^2 + \delta \sum_{j=1}^{J} |\beta_j| \quad [1.10]$$

and $0 \leq \delta \leq 1$ is a parameter for blending the ℓ_1 and ℓ_2 components. A potential drawback of the LASSO is that it tends to somewhat arbitrarily select one predictor from a group of highly correlated predictors. In contrast, using $\delta = 0.5$ in equation [1.9] results in a stronger tendency to select the highly correlated predictors as a group (Hastie and Qian 2016). The corrected AIC can again be used to choose λ in equation [1.9].

A market excess return forecast based on ENet estimation of the multiple predictive regression model in equation [1.3] is given by:

$$\hat{r}^{\text{ENet}}_{t+1|t} = \hat{\alpha}^{\text{ENet}}_{1:t} + \sum_{j=1}^{J} \beta^{\text{ENet}}_{j,1:t} x_{j,t}, \qquad [1.11]$$

where $\hat{\alpha}^{\text{ENet}}_{1:t}$ and $\hat{\beta}^{\text{ENet}}_{j,1:t}$ are the ENet estimates of α and β_j, respectively, for $j = 1, \ldots, J$ in equation [1.3]. Intuitively, we rely on the shrinkage properties of the ENet to generate a market excess return forecast that incorporates information from a potentially large number of predictor variables in a manner that guards against overfitting. Whether the ENet is an effective shrinkage strategy for forecasting the market excess return is ultimately an empirical issue. We investigate this issue in our empirical application in section 1.3[4].

1.2.4. *Combination elastic net*

Incorporating insights from Diebold and Shin (2019), we can also use machine learning techniques to refine the combination forecast in equation [1.7]. A potential drawback to equation [1.7] is that it may "overshrink" the forecast to the prevailing mean, thereby neglecting substantive relevant information in the predictor variables. In an effort to improve the combination forecast by exploiting more of the relevant information in the predictor variables (while still avoiding overfitting), we consider the following Granger and Ramanathan (1984) regression:

$$r_t = \eta + \sum_{j=1}^{J} \theta_j \hat{r}^{(j)}_{t|t-1} + \varepsilon_t, \qquad [1.12]$$

which we estimate via the elastic net to select the most relevant univariate forecasts to include in the combination forecast[5]. Specifically, to construct the combination elastic net (C-ENet) forecast, we first need to define an initial

4 We focus on the results for the ENet in our empirical application in section 1.3, although the results are qualitatively similar for the LASSO.

5 Again, the results are qualitatively similar in our empirical application in section 1.3 if we use the LASSO to estimate equation [1.12].

in-sample estimation period and corresponding holdout out-of-sample period; let t_1 denote the size of the initial in-sample period. We then proceed in three steps:

Step 1 For each predictor variable, we compute recursive univariate predictive regression forecasts based on equation [1.2] over the holdout out-of-sample period:

$$\hat{r}^{(j)}_{s|s-1} = \hat{\alpha}^{(j)}_{1:s-1} + \hat{\beta}^{(j)}_{1:s-1} x_{j,s-1}, \qquad [1.13]$$

for $s = t_1 + 1, \ldots, t$ and $j = 1, \ldots, J$.

Step 2 We estimate the Granger and Ramanathan (1984) regression in equation [1.12] via the ENet over the holdout out-of-sample period:

$$r_s = \eta + \sum_{j=1}^{J} \theta_j \hat{r}^{(j)}_{s|s-1} + \varepsilon_s, \qquad [1.14]$$

for $s = t_1 + 1, \ldots, t$. Let $\mathcal{J}_t \subseteq \{1, \ldots, J\}$ denote the index set of individual univariate predictive regression forecasts selected by the ENet in equation [1.14]. When estimating equation [1.14], we impose the restriction that $\theta_j \geq 0$ for $j = 1, \ldots, J$. This imposes the economically reasonable requirement that a univariate market excess return forecast be positively related to the realized excess return in order to be selected by the ENet in equation [1.14].

Step 3 We compute the C-ENet forecast as:

$$\hat{r}^{\text{C-ENet}}_{t+1|t} = \frac{1}{|\mathcal{J}_t|} \sum_{j \in \mathcal{J}_t} \hat{r}^{(j)}_{t+1|t}, \qquad [1.15]$$

where $|\mathcal{J}_t|$ is the cardinality of $\mathcal{J}_t$ and $\hat{r}^{(j)}_{t+1|t}$ is given by equation [1.2] for $j = 1, \ldots, J$.

The usefulness of the C-ENet approach for capturing the relevant information in numerous predictor variables, in a manner that guards against overfitting, is again ultimately an empirical issue. In our empirical application in section 1.3, we find that the C-ENet approach is indeed an effective strategy for forecasting the market excess return[6].

6 Observe that all the forecasts that we construct only use data available through t to forecast r_{t+1}, so that the forecasts do not entail "look-ahead" bias.

1.3. Empirical application

1.3.1. *Data*

We investigate the performance of the strategies discussed in section 1.2 for forecasting the monthly S&P 500 excess return. Using data available from Amit Goyal's website[7], we measure the excess return as the CRSP value-weighted S&P 500 return in excess of the risk-free return (based on the Treasury bill rate).

We consider 12 plausible predictor variables, which are illustrative of popular predictors used by academics and practitioners alike:

– *Log dividend-price ratio* (*DP*). Log of the 12-month moving sum of S&P 500 dividends minus the log of the S&P 500 price index.

– *Log earnings-price ratio* (*EP*). Log of the 12-month moving sum of S&P 500 earnings minus the log of the S&P 500 price index.

– *Volatility* (*VOL*). We follow Mele (2007) in measuring the annualized volatility for month t as $\sqrt{\frac{\pi}{2}}\sqrt{12}\hat{\sigma}_t$, where $\hat{\sigma}_t = \frac{1}{12}\sum_{s=1}^{12}|r_{t-(s-1)}|$.

– *Treasury bill yield* (*BILL*). Three-month Treasury bill yield minus the 12-month moving average of the three-month Treasury bill yield.

– *Treasury bond yield* (*BOND*). Ten-year Treasury bond yield minus the 12-month moving average of the ten-year Treasury bond yield.

– *Term spread* (*TERM*). Difference in yields on a ten-year treasury bond and a three-month treasury bill.

– *Credit spread* (*CREDIT*). Difference in yields on a AAA-rated corporate bond and a ten-year treasury bond.

– *Inflation* (*PPIG*). Producer price index (PPI) inflation rate.

– *Industrial production growth* (*IPG*). Growth rate of industrial production.

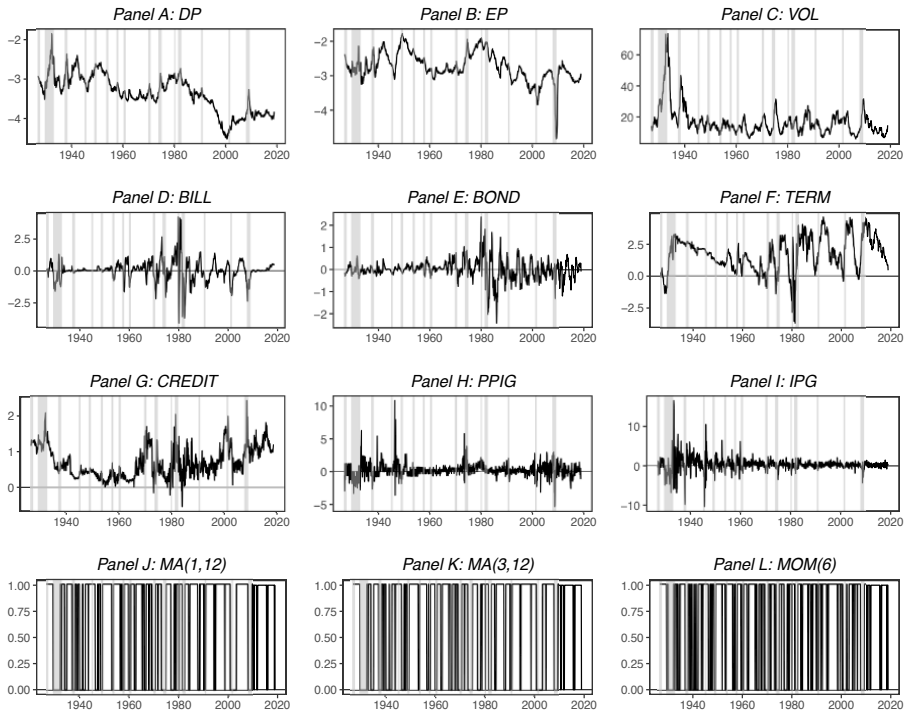
– *MA(1,12) technical signal* [*MA(1,12)*]. An indicator variable that takes a value of one (zero) if the S&P 500 price index is greater than or equal to (less than) the 12-month moving average of the S&P 500 price index.

– *MA(3,12) technical signal* [*MA(3,12)*]. An indicator variable that takes a value of one (zero) if the three-month moving average of the S&P 500 price

7 http://www.hec.unil.ch/agoyal/.

index is greater than or equal to (less than) the 12-month moving average of the S&P 500 price index.

– *Momentum technical signal* [*MOM(6)*]. An indicator variable that takes a value of one (zero) if the S&P 500 price index is greater than or equal to (less than) its value six months ago.



The figure depicts 12 predictor variables for 1927:01 to 2018:12. The predictor variable definitions are provided in section 1.3.1. Vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.1. *Predictor variables*

The data used to construct the predictor variables are from Amit Goyal's website and the Federal Reserve Bank of St. Louis's Federal Reserve Economic Data (FRED)[8]. We account for the one-month publication lag in *PPIG* and *IPG*. We follow Neely *et al.* (2014) in defining indicator variables to include information from technical signals. Figure 1.1 portrays the 12

8 https://fred.stlouisfed.org/.

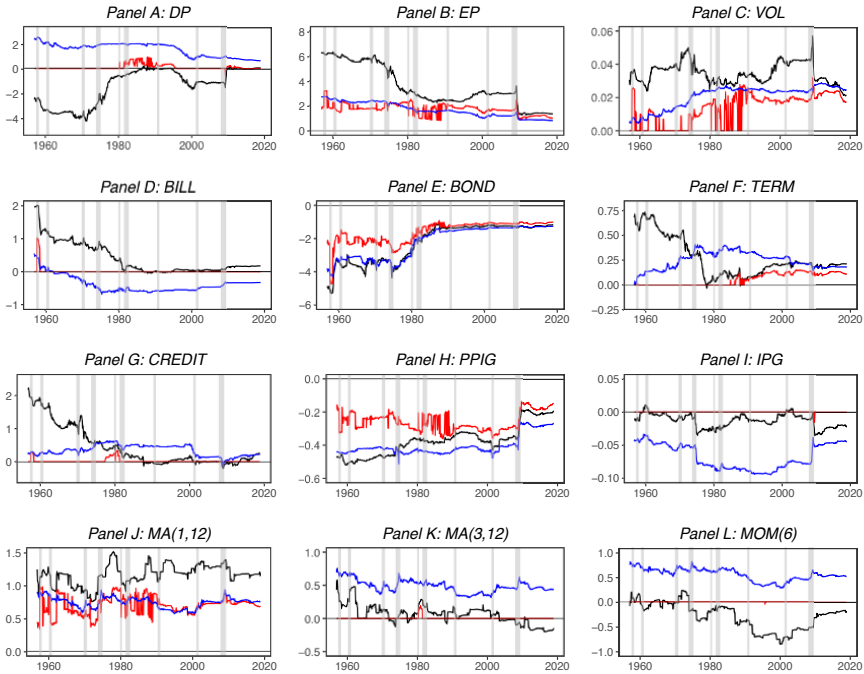
predictor variables for the 1927:01 to 2018:12 sample period. Visually, the predictors represent a variety of information sources.

1.3.2. *Forecasts*

We reserve the first two decades of the sample (1927:01 to 1946:12) as the initial in-sample estimation period. This provides us with an adequate number of observations to reasonably reliably estimate the predictive regression coefficients in equations [1.1] and [1.3]. The initial holdout out-of-sample period for computing the C-ENet forecast covers 1947:01 to 1956:12, so that we evaluate the out-of-sample forecasts for 1957:01 to 2018:12. The out-of-sample forecast evaluation period covers more than six decades, which allows us to analyze return predictability under a variety of economic conditions.

Figure 1.2 depicts the recursive slope coefficient estimates used to compute the predictive regression forecasts in equations [1.2], [1.4] and [1.11]. The black line in each panel delineates the OLS slope coefficient estimates for the multiple predictive regression model in equation [1.3]. The recursive estimates point to problems with the conventional OLS estimates of the multiple predictive regression model slope coefficients – the estimates often have the "wrong" sign (e.g. DP and $BILL$) and reach extreme values, which are manifestations of in-sample overfitting. Overfitting is not surprising when we rely on conventional methods to estimate a relatively high-dimensional predictive regression model in a noisy environment.

The blue line in each panel of Figure 1.2 delineates the recursive OLS slope coefficient estimates for the univariate predictive regression model in equation [1.1]. Compared to the recursive slope coefficient estimates for the multiple predictive regression model, the univariate estimates are generally much more stable. This reflects the increase in estimation precision afforded by the mitigation of multi-collinearity. Of course, the univariate estimates are potentially biased (due to omitted variable bias). However, in light of the bias-efficiency trade-off, the increase in estimation precision can outweigh the cost of the bias for the purpose of out-of-sample forecasting.

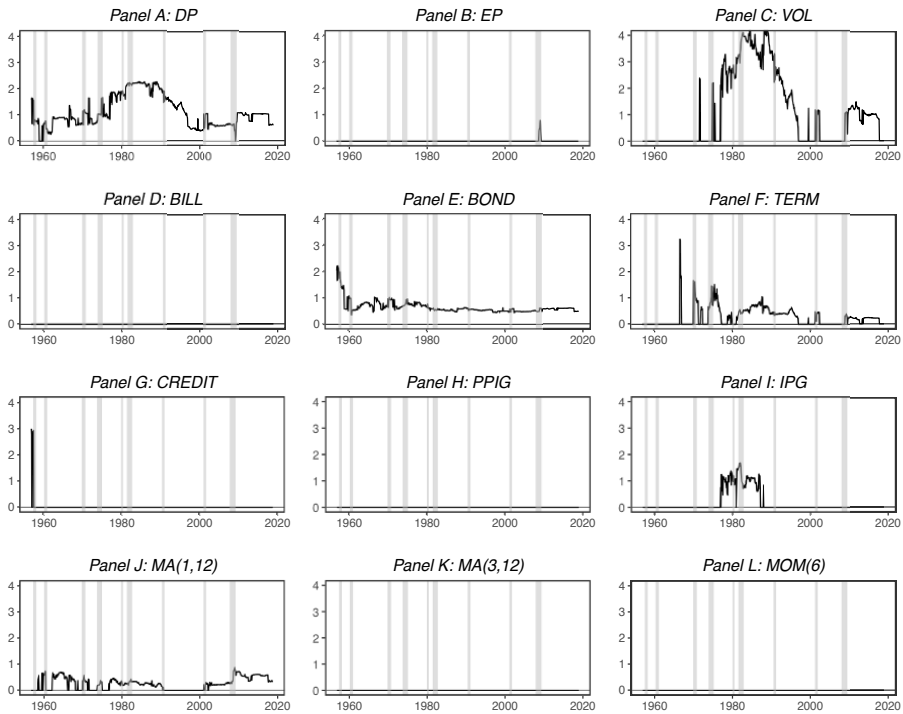


The figure depicts recursive predictive regression model slope coefficient estimates used to compute market excess return forecasts. The black and red lines delineate multiple predictive regression model slope coefficients estimated via ordinary least squares and the elastic net, respectively, for the predictor variable in the panel heading. The blue lines delineate univariate predictive regression model slope coefficients estimated via ordinary least squares. The predictor variable definitions are provided in section 1.3.1. The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.2. *Recursive predictive regression model slope coefficient estimates. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

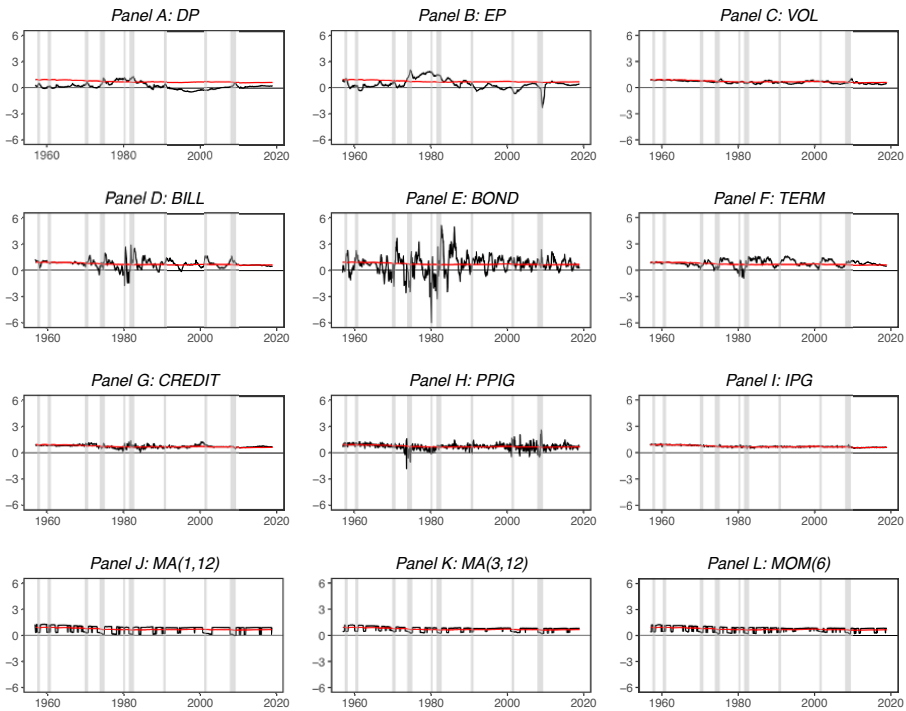
The red line in each panel of Figure 1.2 shows the ENet slope coefficient estimates for the multiple predictive regression model in equation [1.3]. The shrinkage effect of ENet *vis-à-vis* OLS estimation of the multiple regression slope coefficients is clear in Figure 1.2. For some of the predictor variables – such as *BILL*, *IPG*, *MA(3,12)* and *MOM(6)* – the ENet nearly always shrinks the coefficient estimates all the way to zero. The shrinkage induced by ENet estimation of the multiple predictive regression model in equation [1.3] should help at least somewhat in alleviating overfitting.

Figure 1.3 presents the recursive ENet slope coefficient estimates for the Granger and Ramanathan (1984) regression in equation [1.14] used to compute the C-ENet forecast. Recall that the C-ENet forecast is the average of the individual univariate predictive regression forecasts selected by the ENet in equation [1.14]. Figure 1.3 indicates that *BOND* is always selected by the ENet for inclusion in the C-ENet forecast, while *DP* is nearly always selected. *VOL*, *TERM*, and *MA(1,12)* are also frequently selected for inclusion in the C-ENet forecast.



The figure depicts recursive elastic net slope coefficient estimates for a Granger and Ramanathan (1984) regression based on univariate predictive regression forecasts for 12 individual predictor variables. The predictor variable definitions are provided in section 1.3.1. The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.3. *Recursive elastic net Granger and Ramanathan (1984) regression slope coefficient estimates*

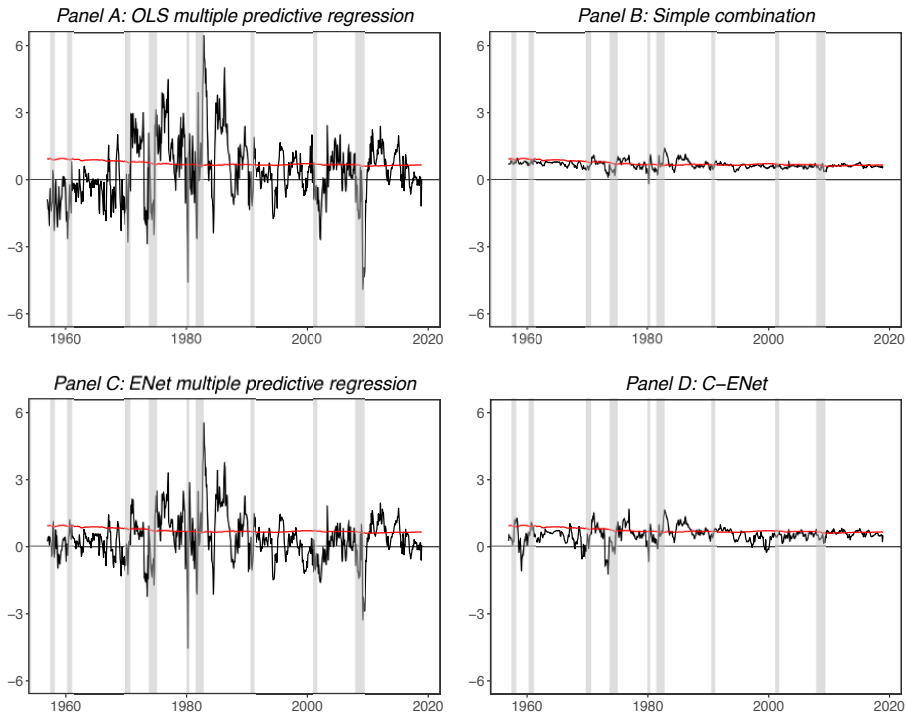


The figure depicts out-of-sample market excess return forecasts (in %) for 1957:01 to 2018:12. The black line in each panel delineates a forecast based on ordinary least squares estimation of a univariate predictive regression model with the predictor variable in the panel heading. The predictor variable definitions are provided in section 1.3.1. The red line in each panel delineates the prevailing mean benchmark forecast. The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.4. *Market excess return forecasts based on individual predictor variables. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Univariate predictive regression forecasts based on equation [1.2] for the 12 individual predictor variables are shown in Figure 1.4, while the forecasts based on multiple predictor variables in equations [1.4], [1.7], [1.11] and [1.15] are presented in Figure 1.5. The red line in each panel delineates the prevailing mean benchmark forecast. A number of the univariate forecasts in Figure 1.4 exhibit distinct behavior over the business cycle. In general, the market excess return forecast lies below the prevailing mean benchmark in the months preceding and early months of a recession; the excess return

forecast then increases, so that it moves above the prevailing mean benchmark in the later months of and months immediately following a recession. The forecasts based on the valuation ratios (*DP* and *EP*) also display some longer swings that persist beyond business-cycle frequencies.



The figure depicts out-of-sample market excess return forecasts (in %) for 1957:01 to 2018:12 based on 12 predictor variables using the method in the panel heading. ENet (C-ENet) stands for elastic net (combination elastic net). The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.5. *Market excess return forecasts based on multiple predictor variables. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Panel A of Figure 1.5 depicts the OLS multiple predictive regression forecast in equation [1.4]. The forecast generally displays the same type of behavior over the business cycle as many of the univariate predictive regression forecasts in Figure 1.4. However, the OLS multiple predictive regression forecast is substantially more volatile and it reaches quite extreme

positive and negative values for numerous months over the out-of-sample evaluation period[9]. This is symptomatic of overfitting. Because it maximizes the fit over the estimation sample, conventional estimation is vulnerable to over-responding to noise in the data, especially in a setting with a low signal-to-noise ratio (such as stock return forecasting).

The simple combination forecast in Panel B of Figure 1.5 is the average of the 12 univariate forecasts in Figure 1.4. As discussed in section 1.2.2, forecast combination exerts a strong shrinkage effect, which is immediately evident from the sharp reduction in forecast volatility as we move from Panel A to Panel B in Figure 1.5.

ENet estimation of the multiple predictive regression model in equation [1.3] provides an alternative means for inducing shrinkage in the forecasts. From Panel C of Figure 1.5, we see that ENet estimation does induce some forecast shrinkage relative to the OLS forecast in Panel A. However, the degree of shrinkage is considerably weaker than that induced by the simple combination forecast in Panel B.

The degree of forecast shrinkage induced by the C-ENet forecast in Panel D of Figure 1.5 falls between that of the simple combination forecast in Panel B and ENet multiple predictive regression forecast in Panel C. By not necessarily averaging across all of the univariate forecasts, the C-ENet forecast is able to respond more strongly to fluctuations in the univariate forecasts selected by the ENet in the Granger and Ramanathan (1984) regression, in equation [1.14]. In this fashion, the C-ENet forecast is designed to mitigate the potential overshrinking induced by the simple combination forecast, while avoiding the complete lack of shrinkage in the OLS multiple predictive regression forecast.

1.3.3. *Statistical gains*

Next, we assess the statistical accuracy of the forecasts in Figures 1.4 and 1.5 in terms of mean squared forecast error (MSFE). To this end, we compute the Campbell and Thompson (2008) out-of-sample R^2 statistic:

$$R^2_{\text{OS}} = 1 - \frac{\sum_{s=t_2+1}^{T} \left(\hat{e}_{s|s-1}\right)^2}{\sum_{s=t_2+1}^{T} \left(\hat{e}^{\text{PM}}_{s|s-1}\right)^2}, \qquad [1.16]$$

9 Note that the forecasts are not annualized in Figures 1.4 and 1.5.

where t_2 is the last observation for the initial holdout out-of-sample period, T is the total number of available observations, $\hat{e}_{s|s-1} = r_s - \hat{r}_{s|s-1}$ generically denotes a competing forecast error, and $\hat{e}^{\text{PM}}_{s|s-1} = r_s - \hat{r}^{\text{PM}}_{s|s-1}$ is the prevailing mean forecast error. The R^2_{OS} statistic, which is akin to the familiar in-sample R^2 statistic, measures the proportional reduction in MSFE for a competing forecast *vis-à-vis* the prevailing mean benchmark forecast. Of course, because monthly stock returns inherently contain a limited predictable component, the R^2_{OS} statistic will be "small"[10]. Nevertheless, Campbell and Thompson (2008) argue that a monthly R^2_{OS} statistic of approximately 0.5%, indicates an economically significant degree of market excess return predictability.

To gauge whether a competing forecast provides a statistically significant improvement in MSFE relative to the prevailing mean benchmark forecast, we use the Clark and West (2007) adjusted version of the familiar Diebold and Mariano (1995) and West (1996) statistic. The latter is less informative for comparing forecasts from nested models (Clark and McCracken 2001; McCracken 2007). In particular, we use the Clark and West (2007) MSFE-adj statistic to test the null hypothesis that the prevailing mean MSFE is less than or equal to the competing MSFE, against the alternative that the competing MSFE is less than the prevailing mean MSFE.

Panel A of Table 1.1 reports R^2_{OS} statistics for the univariate predictive regression forecasts. The R^2_{OS} statistics are negative for the majority of the predictor variables, so that most of the forecasts fail to outperform the prevailing mean benchmark in terms of MSFE. The univariate forecasts based on *VOL*, *BILL*, *BOND*, *TERM*, and *MA(1,12)* produce positive R^2_{OS} statistics, and the MSFE-adj statistics indicate that each of the five forecasts provides a statistically significant reduction in MSFE *vis-à-vis* the prevailing mean benchmark (at the 10% level or better)[11]. Among the positive R^2_{OS} statistics, only that for *BOND* is above the Campbell and Thompson (2008) threshold of 0.5%. Furthermore, we cannot know *a priori* which of the 12 predictor variables in Panel A will perform the best.

10 Indeed, if it is "large", it is likely too good to be true!

11 Despite having a negative R^2_{OS} statistic, the MSFE-adj statistic for *DP* is significant. Although this result may seem surprising, it can occur when comparing nested forecasts (Clark and West 2007; McCracken 2007).

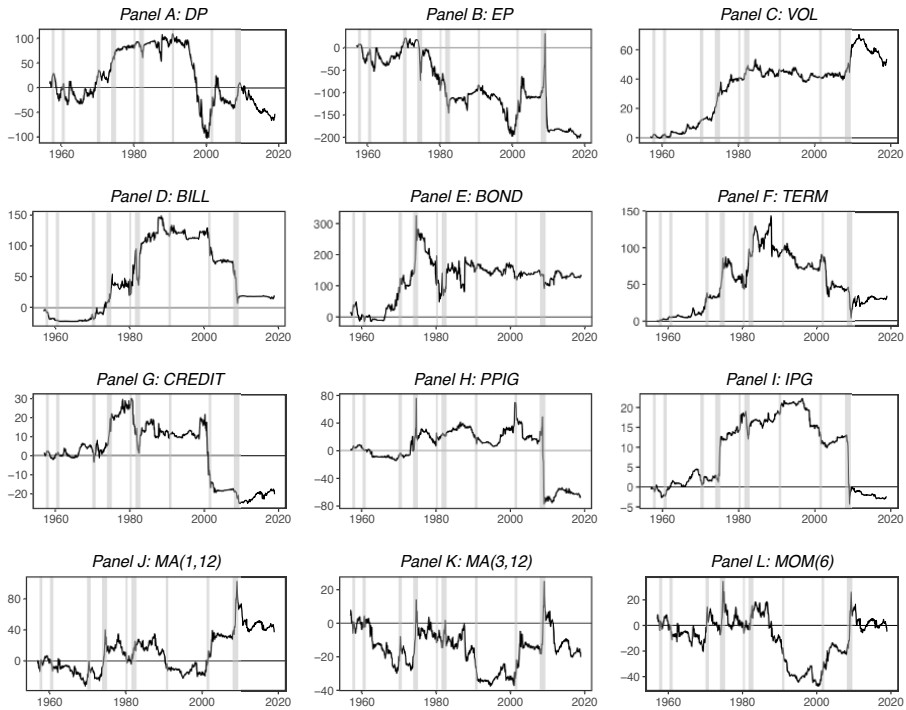
(1) Forecast	(2) R^2_{OS}	(3) MSFE-adj	(4) Forecast	(5) R^2_{OS}	(6) MSFE-adj
Panel A: Individual predictor variables					
DP	–0.40%	1.81**	*CREDIT*	–0.15%	–0.16
EP	–1.47%	0.79	*PPIG*	–0.50%	0.18
VOL	0.42%	2.58***	*IPG*	–0.02%	–0.04
BILL	0.15%	1.63*	*MA(1,12)*	0.28%	1.38*
BOND	1.04%	3.37***	*MA(3,12)*	–0.15%	0.32
TERM	0.26%	1.59*	*MOM(6)*	–0.04%	0.71
Panel B: Multiple predictor variables					
OLS multiple predictive regression	–4.33%	2.73***	ENet multiple predictive regression	0.22%	3.11***
Simple combination	1.11%	3.70***	C-ENet	2.12%	4.05***

The table reports out-of-sample R^2 (R^2_{OS}) statistics for monthly market excess return forecasts for 1957:01 to 2018:12. The out-of-sample R^2 statistic is the percent reduction in mean squared forecast error (MSFE) for a competing forecast *vis-á-vis* the prevailing mean benchmark forecast. The competing forecasts in Panel A are based on univariate predictive regression models estimated via ordinary least squares. The predictor variable definitions in Panel A are provided in section 1.3.1. The competing forecasts in Panel B use all 12 predictor variables. The OLS (ENet) multiple predictive regression forecast is based on a multiple predictive regression model with all 12 predictor variables estimated via ordinary least squares (the elastic net). The simple combination forecast is the average of the individual univariate predictive regression forecasts in Panel A. The C-ENet forecast is an average of the individual univariate predictive regression forecasts in Panel A selected by the elastic net in a Granger and Ramanathan (1984) regression. MSFE-adj is the Clark and West (2007) statistic for testing the null hypothesis that the benchmark MSFE is less than or equal to the competing MSFE against the alternative hypothesis that the benchmark MSFE is greater than the competing MSFE; *, ** and *** indicate significance at the 10%, 5% and 1% levels, respectively.

Table 1.1. *Forecast accuracy*

To get a sense of the performance of the individual univariate forecasts over time, Figure 1.6 shows the cumulative differences in squared forecast errors for the prevailing mean benchmark relative to each univariate forecast (Goyal and Welch 2003, 2008). The cumulative differences in Figure 1.6 make it straightforward to determine whether a competing forecast is more accurate than the prevailing mean benchmark for any subsample. We simply compare the height of the curve at the start and end of the subsample. If the curve is higher (lower) at the end, then the competing forecast has a lower

(higher) MSFE than the prevailing mean benchmark over the subsample. A competing forecast that provides out-of-sample gains on a consistent basis will thus have a predominantly positively sloped curve, while a steeply negatively sloped segment indicates an episode of severe underperformance.



The figure depicts cumulative differences in squared forecast errors between benchmark and competing out-of-sample market excess return forecasts for 1957:01 to 2018:12. The competing forecast is based on ordinary least squares estimation of a univariate predictive regression model with the predictor variable in the panel heading; the benchmark forecast is the prevailing mean. The predictor variable definitions are provided in section 1.3.1. The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.6. *Cumulative differences in squared forecast errors for market excess return forecasts based on individual predictor variables*

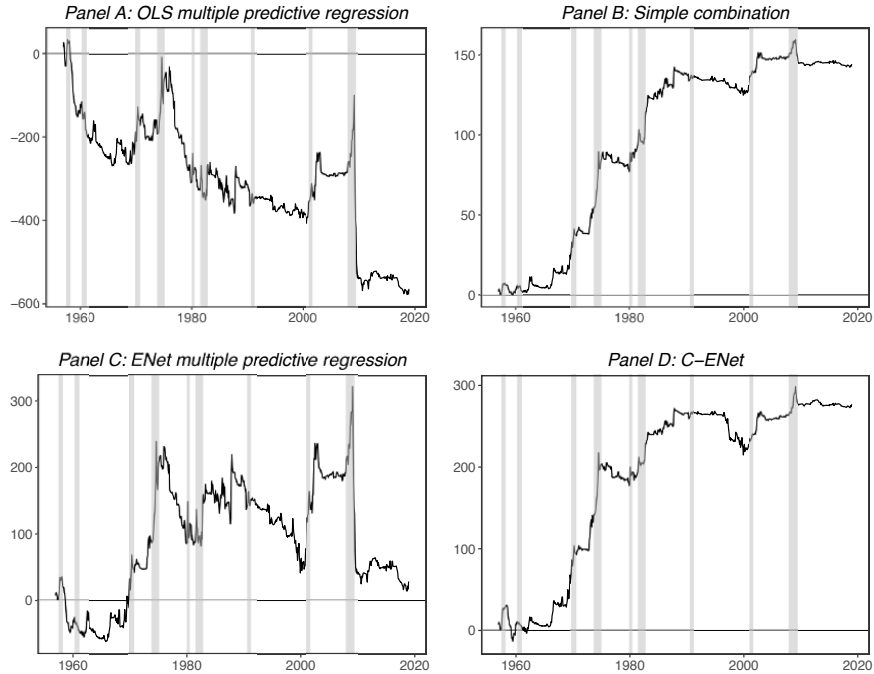
With the exception of *VOL*, the univariate forecasts in Figure 1.6 fail to provide accuracy gains on a reasonably consistent basis over time. Although

there are times when the univariate forecasts substantively outperform the prevailing mean benchmark, there are also periods when they perform much worse than the naive benchmark. Overall, Figure 1.6 is reminiscent of the findings in Goyal and Welch (2008).

Panel B of Table 1.1 reports R^2_{OS} statistics for the four forecasts that incorporate information from multiple predictor variables, while Figure 1.7 depicts the cumulative differences in squared forecast errors for the prevailing mean benchmark relative to the four forecasts. The R^2_{OS} statistic for the OLS multiple predictive regression forecast is -4.33% in Panel B of Table 1.1, so that the forecast is substantially less accurate than the prevailing mean benchmark over the 1957:01 to 2018:12 evaluation period. Panel A of Figure 1.7 shows that the forecast exhibits prolonged periods of underperformance relative to the naive benchmark. Overall, the OLS multiple predictive regression forecast apparently suffers from considerable overfitting.

The R^2_{OS} statistic for the simple combination forecast is 1.11% in Panel B of Table 1.1 (and its MSFE-adj statistic is significant at the 1% level), which is larger than all the R^2_{OS} statistics for the univariate predictive regression forecasts in Panel A. In addition, Panel B of Figure 1.7 reveals that the simple combination forecast outperforms the prevailing mean benchmark on a reasonably consistent basis over time, as the curve is predominantly positively sloped and displays only limited segments with a negative slope. The strong shrinkage induced by the simple combination forecast is apparently useful for incorporating information from multiple predictor variables in a manner that guards against overfitting.

Although the ENet multiple predictive regression forecast also induces shrinkage, recall from Figure 1.5 that the degree of shrinkage appears limited. The R^2_{OS} statistic for the ENet forecast is positive but quite close to zero (0.22%) in Panel B of Table 1.1. Furthermore, Panel C of Figure 1.7 indicates that the ENet forecast fails to outperform the prevailing mean benchmark consistently over time. Indeed, there are a number of segments where the ENet forecast severely underperforms the prevailing mean. It thus seems that the degree of shrinkage induced by the ENet multiple predictive regression forecast is too weak to effectively guard against overfitting when forecasting the market excess return.



The figure depicts cumulative differences in squared forecast errors between benchmark and competing out-of-sample market excess return forecasts for 1957:01 to 2018:12. The competing forecast is based on 12 predictor variables using the method in the panel heading; the benchmark forecast is the prevailing mean. ENet (C-ENet) stands for elastic net (combination elastic net). The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.7. *Cumulative differences in squared forecast errors for market excess return forecasts based on multiple predictor variables*

The simple combination forecast exerts a strong – perhaps too strong – shrinkage effect, while the ENet multiple predictive regression forecast induces insufficient shrinkage. Again recalling Figure 1.5, the degree of shrinkage induced by the C-ENet forecast falls between that induced by the simple combination and ENet forecasts. The R^2_{OS} statistic for the C-ENet forecast is a sizable 2.12% in Panel B of Table 1.1 (and its MSFE-adj statistic is significant at the 1% level). The ENet forecast provides the largest improvement in MSFE among all the competing forecasts relative to the prevailing mean benchmark, and its R^2_{OS} statistic is nearly double that for the

simple combination forecast. Panel C of Figure 1.7 confirms that the C-ENet forecast outperforms the prevailing mean quite consistently over time[12].

Overall, the C-ENet forecast appears close to a "Goldilocks" shrinkage technique. It induces stronger shrinkage than the ENet forecast, so that it better guards against overfitting. At the same time, it exerts a weaker shrinkage effect than the simple combination forecast, allowing it to incorporate more information from the most relevant univariate forecasts to further improve out-of-sample performance. Recall from Figure 1.3 that the ENet often selects the univariate predictive regression forecasts based on *DP*, *VOL*, *BOND*, *TERM*, and *MA(1,12)* in the Granger and Ramanathan (1984) regression. This judicious real-time selection of relevant predictor variables to include in the combination forecast demonstrates the value of machine learning for improving out-of-sample market excess return forecasts.

1.3.4. *Economic gains*

In addition to the gains in forecast accuracy documented in section 1.3.3, we next show that the C-ENet forecast provides substantial economic gains for an investor in an asset allocation context. Specifically, we consider a mean-variance investor who allocates across risky equities and risk-free Treasury bills each month. The investor's objective function is given by:

$$\arg\max_{w_{t+1|t}} w_{t+1|t}\hat{r}_{t+1|t} - 0.5\gamma w_{t+1|t}^2\hat{\sigma}_{t+1|t}^2, \qquad [1.17]$$

where γ is the coefficient of relative risk aversion, $w_{t+1|t}$ $(1 - w_{t+1|t})$ is the investor's allocation to equities (risk-free bills) in month $t + 1$ and $\hat{r}_{t+1|t}$ $(\hat{\sigma}_{t+1|t}^2)$ is the market excess return point (variance) forecast used by the investor. The well-known solution to equation [1.17] takes the form[13]:

$$w_{t+1|t}^* = \left(\frac{1}{\gamma}\right)\left(\frac{\hat{r}_{t+1|t}}{\hat{\sigma}_{t+1|t}^2}\right). \qquad [1.18]$$

12 The out-of-sample gains in Figure 1.7 are often particularly evident during business-cycle recessions. This is a stylized fact in the literature on market excess return predictability (e.g. Rapach *et al*. 2010; Henkel *et al*. 2011; Rapach and Zhou 2013).

13 To prevent what could be construed as impractical allocations, we impose the restriction that $-1 \leq w_{t+1|t} \leq 2$. As shown in Panel A of Figure 1.8, the constraint is rarely binding.

To measure the economic value of return predictability to the investor, we first analyze portfolio performance when the investor relies on the prevailing mean forecast – which ignores return predictability – to determine the optimal equity allocation in equation [1.18]. We then analyze portfolio performance when the investor instead uses the C-ENet forecast to select the optimal allocation. In both cases, the investor uses the sample variance computed over a 60-month rolling window to forecast the variance[14]. We assume that $\gamma = 5$; the results are qualitatively similar for reasonable alternative values for γ.

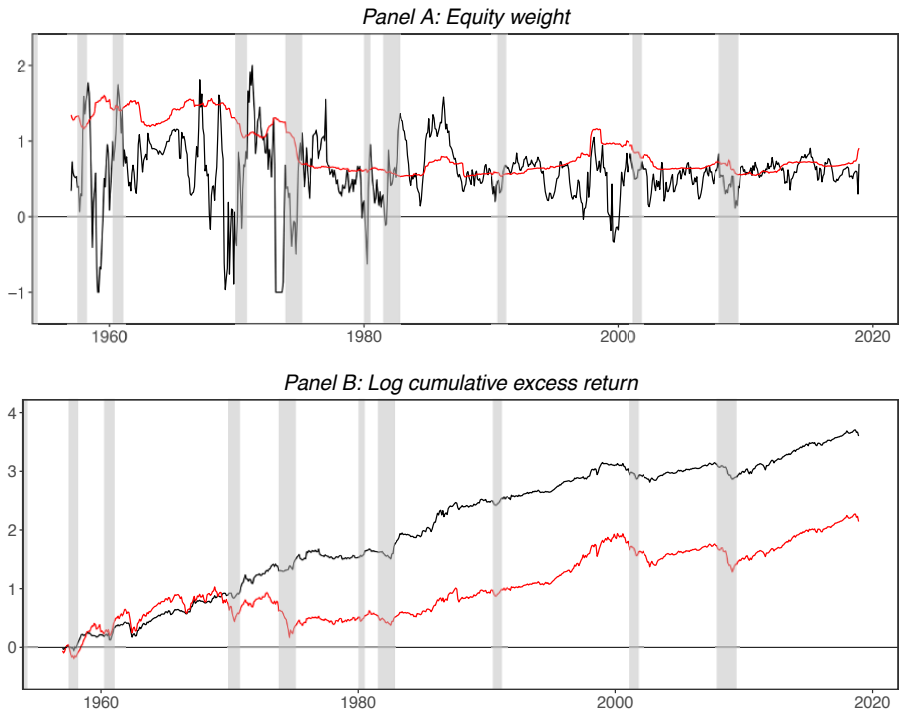
When the investor relies on the prevailing mean benchmark forecast, the optimal portfolio earns an annualized average excess return of 4.30% for the 1957:01 to 2018:12 forecast evaluation period. Together with an annualized volatility of 12.87%, the portfolio's annualized Sharpe ratio is 0.33. If the investor uses the C-ENet forecast in lieu of the prevailing mean in equation [1.18], then the optimal portfolio yields an annualized average excess return of 6.29% and volatility of 9.76%. These translate into a substantive annualized Sharpe ratio of 0.64, which is nearly twice as large as that for the portfolio based on the prevailing mean[15]. Moreover, the gain in certainty equivalent return (CER) indicates that the investor would be willing to pay a hefty annualized management fee of 375 bps to switch from the prevailing mean to the C-ENet forecast.

The red and black lines in Panel A of Figure 1.8 depict the equity allocations for the optimal portfolios based on the prevailing mean and C-ENet forecasts, respectively. There are numerous months when the C-ENet forecast leads to a markedly different allocation than the prevailing mean benchmark. Such reallocations appear quite valuable to the investor, as they substantially improve portfolio performance. This is further illustrated in Panel B of Figure 1.8, which shows the log cumulative excess returns for the two portfolios. For example, the portfolio based on the C-ENet forecast typically suffers smaller drawdowns than the portfolio based on the prevailing

14 In this chapter, we focus on the ability of machine learning techniques to improve expected return forecasts. It would be interesting in future research to explore the usefulness of machine learning techniques for improving forecasts of return volatilities and correlations.

15 The annualized Sharpe ratio for the market excess return for 1957:01 to 2018:12 is 0.42, so that the optimal portfolio based on the C-ENet forecast delivers a Sharpe ratio that is over 50% larger than that for the market portfolio.

mean benchmark; indeed, the maximum drawdown for the former is only half as large as that for the latter (29% and 58%, respectively).



The black and red line in Panel A delineate the equity weight for a mean-variance investor with a coefficient of relative risk aversion of five who uses the combination elastic net (prevailing mean) forecast when allocating between equities and risk-free Treasury bills for 1957:01 to 2018:12. The lines in Panel B show the corresponding log cumulative excess returns for the two portfolios. The vertical bars delineate business-cycle recessions as dated by the National Bureau of Economic Research.

Figure 1.8. *Equity allocations and cumulative excess returns. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The optimal portfolio based on the C-ENet forecast also generates a higher Sharpe ratio and annualized CER gain than the optimal portfolios based on any of the 12 univariate predictive regression forecasts, as well as the OLS and ENet multiple predictive regression and simple combination forecasts[16].

16 The complete results are available upon request from the authors.

In sum, the C-ENet forecast delivers the best overall performance in terms of statistical accuracy and investor value.

1.4. Cross-sectional return forecasts

In this section, we outline the cross-sectional return forecasting procedures in Han *et al*. (2019) which parallel the time-series strategies in section 1.2[17]. As Lewellen (2015) argues, the appropriate target is the cross-sectional dispersion in firm returns (and not the cross-sectional average return *per se*); we can simply adjust all of the cross-sectional return forecasts up or down as needed to reflect our forecast of the overall market return.

Consider the following cross-sectional multiple regression model for month t:

$$r_{i,t} = \alpha_t + \sum_{j=1}^{J} \beta_{j,t} z_{i,j,t-1} + \varepsilon_{i,t}, \qquad [1.19]$$

for $i = 1, \ldots, I_t$, where $z_{i,j,t}$ is the jth characteristic for firm i in month t and I_t is the number of firm observations available in month t. Analogous to equation [1.4], a conventional cross-sectional return forecast based on equation [1.19] is given by:

$$\hat{r}^{\text{OLS}}_{i,t+1|t} = \hat{\alpha}^{\text{OLS}}_t + \sum_{j=1}^{J} \hat{\beta}^{\text{OLS}}_{j,t} z_{i,j,t}, \qquad [1.20]$$

where $\hat{\alpha}^{\text{OLS}}_t$ and $\hat{\beta}^{\text{OLS}}_{j,t}$ are the OLS estimates of α_t and $\beta_{j,t}$, respectively, for $j = 1, \ldots, J$ in equation [1.19]. The literature on cross-sectional returns has investigated many characteristics (e.g. Harvey *et al*. 2016), so that the number of plausible predictors J to include in equation [1.19] is quite large. As in the time-series context, conventional estimation of equation [1.19] is prone to overfitting.

It is common to smooth the OLS coefficient estimates in equation [1.20] over time (e.g. Lewellen 2015; Green *et al*. 2017). However, this appears

17 See Han *et al*. (2019) for further details on the construction of the cross-sectional stock return forecasts discussed in this section.

inadequate for mitigating overfitting. To see this, we use data for 84 firm characteristics starting in 1980:01 to forecast cross-sectional stock returns[18]. The firm characteristics are similar to those used in Lewellen (2015); Green *et al*. (2017) and Freyberger *et al*. (2019). We compute out-of-sample cross-sectional stock returns for 1990:01 to 2018:06 using equation [1.20], although we smooth the OLS coefficient estimates over time using an expanding window before forming the forecasts.

Lewellen (2015) provides an informative predictive slope to assess the ability of a forecast to track cross-sectional expected returns. We compute the predictive slope in two steps. In the first step, we estimate a cross-sectional version of a Mincer and Zarnowitz (1969) regression for month t:

$$r_{i,t} = \phi_t + \psi_t \hat{r}_{i,t|t-1} + \varepsilon_{i,t}, \qquad [1.21]$$

where $\hat{r}_{i,t|t-1}$ generically denotes a cross-sectional return forecast. We estimate ψ_t via OLS for each month over the forecast evaluation period. In the second step, we compute the time-series average of the monthly cross-sectional slope coefficient estimates in equation [1.21]; we denote the average predictive slope estimate by $\hat{\psi}$. As discussed by Lewellen (2015), $\psi = 1$ indicates that the forecasts are unbiased with respect to the cross-sectional dispersion in expected returns: a percentage point increase in the forecast corresponds to a percentage point increase in the realized return on average. If $\psi < 1$ then the cross-sectional forecasts are characterized by overfitting, because a percentage point increase in the forecast corresponds to a less than percentage point increase in the realized return on average. Alternatively, $\psi > 1$ signals that the forecasts are conservative in that a percentage point increase in the forecast coincides with a more than percentage point increase in the actual return on average.

For the OLS multiple regression forecast in equation [1.20], the $\hat{\psi}$ estimate is 0.10 while the average cross-sectional R^2 statistic in equation [1.21] is 0.61%. Based on its standard error, the $\hat{\psi}$ estimate is significantly greater than zero. However, it is also significantly below unity, so that the conventional OLS forecast exhibits significant cross-sectional overfitting, as anticipated.

18 To minimize the effects of micro-cap stocks, we omit stocks with market capitalization below the NYSE median.

Analogous to equation [1.7], Han *et al.* (2019) propose a simple combination strategy for forecasting cross-sectional returns:

$$\hat{r}^{\mathrm{C}}_{i,t+1|t} = \frac{1}{J}\sum_{j=1}^{J} \hat{r}^{(j)}_{i,t+1|t}, \qquad [1.22]$$

where

$$\hat{r}^{(j)}_{i,t+1|t} = \hat{\alpha}^{(j)}_t + \hat{\beta}^{(j)}_t z_{i,j,t}, \qquad [1.23]$$

and $\hat{\alpha}^{(j)}_t$ and $\hat{\beta}^{(j)}_t$ are the OLS estimates of the intercept and slope coefficients, respectively, in the following cross-sectional univariate regression model:

$$r_{i,t} = \alpha_t + \beta_t z_{i,j,t-1} + \varepsilon_{i,t}, \qquad [1.24]$$

for $j = 1, \ldots, J$. Again like the time-series case, the simple combination forecast in equation [1.22] exerts a strong shrinkage effect[19]. The $\hat{\psi}$ estimate is 2.27 for the combination forecast (with an average cross-sectional R^2 of 4.85%), which is significantly greater than both zero and unity. The simple combination approach thus appears to overshrink the forecast, rendering it overly conservative on average.

Han *et al.* (2019) recommend refining the simple combination forecast in equation [1.22] using machine learning techniques. Specifically, they suggest selecting the individual characteristics to include in the combination forecast by using the LASSO or ENet to estimate a cross-sectional Granger and Ramanathan (1984) regression, that relates month-t realized returns to the univariate forecasts in equation [1.23]. The cross-sectional C-ENet forecast is given by:

$$\hat{r}^{\text{C-ENet}}_{i,t+1|t} = \frac{1}{|\mathcal{J}_t|}\sum_{j\in\mathcal{J}_t} \hat{r}^{(j)}_{i,t+1|t}, \qquad [1.25]$$

19 Observe that we do not smooth the OLS coefficient estimates over time in equation [1.23] when forming the cross-sectional forecast as the simple combination forecast already induces strong shrinkage.

where $\mathcal{J}_t \subseteq \{1, \ldots, J\}$ is the index set of cross-sectional univariate forecasts selected by the ENet in the month-t Granger and Ramanathan (1984) regression. As in the time-series case, the refinement proves efficacious: the $\hat{\psi}$ estimate for the C-ENet forecast is 1.25 (with an average cross-sectional R^2 statistic of 3.65%) which is significantly greater than zero but insignificantly different from unity.

Han *et al*. (2019) also develop a cross-sectional forecast that blends the conventional OLS multiple regression and C-ENet forecasts using the notion of forecast encompassing (Harvey *et al*. 1998). The $\hat{\psi}$ estimate for their encompassing C-ENet forecast is 1.10 (with an average cross-sectional R^2 statistic of 2.69%) which is even closer to unity. The $\hat{\psi}$ estimate is significantly greater than zero and insignificantly different from unity. In an extensive empirical application involving more than 100 firm characteristics, Han *et al*. (2019) show that their encompassing C-ENet approach provides the most accurate forecasts to date of the cross-sectional dispersion in expected stock returns.

1.5. Conclusion

Extending the cross-sectional return forecasting procedures developed by Han *et al*. (2019), this chapter introduces some new machine learning methods for time-series stock return forecasting. Our empirical application focuses on forecasting the US market excess return, a central issue in finance. Despite evidence of in-sample predictability, Goyal and Welch (2008) show that conventional forecasts based on many popular predictor variables from the literature fail to provide out-of-sample gains on a consistent basis over time. Using simple forecast combination, Rapach *et al*. (2010) are seemingly the first to provide evidence of consistent out-of-sample market excess return predictability. The methods proposed in this chapter use the elastic net to refine the simple combination forecast and we find that the elastic net refinement, embodied in our combination elastic net forecast, indeed generates substantive further improvements in out-of-sample market excess return predictability.

Machine learning techniques are often criticized for being "black boxes". However, by performing variable selection, the elastic net (and LASSO) is a machine learning technique that facilitates economic interpretation. In our

empirical application, our combination elastic net approach consistently identifies the dividend-price ratio, volatility, Treasury bond yield, term spread and a popular technical signal, as relevant market excess return predictors. The identification of the most relevant out-of-sample market excess return predictors from among the plethora of predictors from the literature provides a useful guide for researchers in constructing theoretical asset pricing models. As analyzed by Han *et al*. (2019) in a cross-sectional context, the combination elastic net approach can also be used to identify the most relevant firm characteristics for explaining cross-sectional expected returns. More generally, since countless questions in asset pricing and corporate finance are related to either time-series or cross-sectional return forecasting, the combination elastic net approach discussed in this chapter should provide a valuable resource for researchers and practitioners alike.

1.6. Acknowledgements

We thank Emmanuel Jurczenko, the editor, for helpful comments on an earlier draft.

1.7. References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In Petrov, B.N. and Csaki, F. (eds). *Proceedings of the 2nd International Symposium on Information Theory*, Akadémiai Kiadó, Budapest, 267–281.

Bates, J.M. and Granger, C.W.J. (1969). The combination of forecasts. *Journal of the Operational Research Society*, 20(4), 451–468.

Campbell, J.Y. and Thompson, S.B. (2008). Predicting excess stock returns out of sample: Can anything beat the historical average? *Review of Financial Studies*, 21(4), 1509–1531.

Chinco, A., Clark-Joseph, A.D., and Ye, M. (2019). Sparse signals in the cross-section of returns. *Journal of Finance*, 74(1), 449–492.

Clark, T.E. and McCracken, M.W. (2001). Tests of equal forecast accuracy and ecompassing for nested models. *Journal of Econometrics*, 105(1), 85–110.

Clark, T.E. and West, K.D. (2007). Approximately normal tests for equal predictive accuracy in nested models. *Journal of Econometrics*, 138(1), 291–311.

Diebold, F.X. and Mariano, R.S. (1995). Comparing predictive accuracy. *Journal of Business and Economic Statistics*, 13(3), 253–263.

Diebold, F.X. and Shin, M. (2019). Machine learning for regularized survey forecast combination: Partially-egalitarian lasso and its derivatives. *International Journal of Forecasting*, 35(4), 1679–1691.

Flynn, C.J., Hurvich, C.M., and Simonoff, J.S. (2013). Efficiency for regularization parameter selection in penalized likelihood estimation of misspecified models. *Journal of the American Statistical Association*, 108(503), 1031–1043.

Freyberger, J., Neuhierl, A., and Weber, M. (2019). Dissecting characteristics nonparametrically. *Review of Financial Studies*.

Goyal, A. and Welch, I. (2003). Predicting the equity premium with dividend ratios. *Management Science*, 49(5), 639–654.

Goyal, A. and Welch, I. (2008). A comprehensive look at the empirical performance of equity premium prediction. *Review of Financial Studies*, 21(4), 1455–1508.

Granger, C.W.J. and Ramanathan, R. (1984). Improved methods of combining forecasts. *Journal of Forecasting*, 3(2), 197–204.

Green, J., Hand, J.R.M., and Zhang, X.F. (2017). The characteristics that provide independent information about average U.S. monthly stock returns. *Review of Financial Studies*, 30(12), 4389–4436.

Gu, S., Kelly, B.T., and Xiu, D. (2019). Empirical asset pricing via machine learning. *Review of Financial Studies*.

Han, Y., He, A., Rapach, D.E., and Zhou, G. (2019). Firm characteristics and expected stock returns. Working Paper.

Harvey, C.R., Liu, Y., and Zhu, H. (2016).... and the cross-section of expected returns. *Review of Financial Studies*, 29(1), 5–68.

Harvey, D.I., Leybourne, S.J., and Newbold, P. (1998). Tests for forecast encompassing. *Journal of Business and Economic Statistics*, 16(2), 254–259.

Hastie, T. and Qian, J. (2016). Glmnet vignette. Working Paper.

Henkel, S.J., Martin, J.S., and Nadari, F. (2011). Time-varying short-horizon predictability. *Journal of Financial Economics*, 99(3), 560–580.

Hoerl, A.E. and Kennard, R.W. (1970). Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12(1), 69–82.

Hurvich, C.M. and Tsai, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika*, 76(2), 297–307.

Ke, Z.T., Kelly, B.T., and Xiu, D. (2019). Predicting returns with text data. Working Paper.

Kozak, S., Nagel, S., and Santosh, S. (2019). Shrinking the cross section. *Journal of Financial Economics*.

Lewellen, J. (2015). The cross-section of expected stock returns. *Critical Finance Review*, 4(1), 1–44.

Loughran, T. and McDonald, B. (2011). When is a liability not a liability? Textual analysis, dictionaries and 10-Ks. *Journal of Finance*, 66(1), 35–65.

McCracken, M.W. (2007). Asymptotics for out of sample tests of Granger causality. *Journal of Econometrics*, 140(2), 719–752.

Mele, A. (2007). Asymmetric stock market volatility and the cyclical behavior of expected returns. *Journal of Financial Economics*, 86(2), 446–478.

Mincer, J.A. and Zarnowitz, V. (1969). The evaluation of economic forecasts. In Mincer, J.A. (ed.). *Economic Forecasts and Expectations: Analysis of Forecasting Behavior and Performance*, National Bureau of Economic Research. Columbia University Press, New York.

Neely, C.J., Rapach, D.E., Tu, J., and Zhou, G. (2014). Forecasting the equity risk premium: The role of technical indicators. *Management Science*, 60(7), 1772–1791.

Rapach, D.E., Strauss, J.K., and Zhou, G. (2010). Out-of-sample equity premium prediction: Combination forecasts and links to the real economy. *Review of Financial Studies*, 23(2), 821–862.

Rapach, D.E., Strauss, J.K., and Zhou, G. (2013). International stock return predictability: What is the role of the United States? *Journal of Finance*, 68(4), 1633–1662.

Rapach, D.E. and Zhou, G. (2013). Forecasting stock returns. In Elliott, G. Timmermann, A. (eds). *Handbook of Economic Forecasting*, Volume 2A. Elsevier, Amsterdam.

Tetlock, P.C. (2007). Giving content to investor sentiment: The role of media in the stock market. *Journal of Finance*, 62(3), 1139–1168.

Tibshirani, R. (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267–288.

Timmermann, A. (2006). Forecast combinations. In Elliott, G., Granger, C.W.J., Timmermann, A. (eds). *Handbook of Economic Forecasting*, Volume 1. Elsevier, Amsterdam.

West, K.D. (1996). Asymptotic inference about predictive ability. *Econometrica*, 64(5), 1067–1084.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 301–320.

2

In Search of Return Predictability: Application of Machine Learning Algorithms in Tactical Allocation

This chapter conducts a horse race among different machine learning (ML) algorithms and other conventional benchmarks used in the literature for bond–equity tactical asset allocation. Our implementation consists of predicting future stock market direction using publicly available data and open-source software. Our findings suggest that an investor can enhance his or her welfare significantly by deploying the publicly available ML algorithms versus commonly used benchmarks. Our chapter contributes to the literature on asset allocation under estimation risk and to the application of ML, in search of the optimal bond–equity portfolio.

2.1. Introduction

The viability of machine learning (ML) algorithms as a decision tool for rule-based investment is the subject of an ongoing debate (Gu *et al*. 2018 and López de Prado 2018). The low signal-to-noise ratio along with evolving market conditions constitute two of the major obstacles for ML algorithms to map financial data into consistently outperforming automated financial decisions (Segal 2019). The ongoing nature of the debate is fueled by the emergence of new algorithms, changing market conditions and the increasing availability of data sources. A definite answer to the practical usefulness of machine learning algorithms is thus elusive.

Chapter written by Kris BOUDT, Muzafer CELA and Majeed SIMAAN.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Over the last few years, machine learning algorithms have shown an enormous capability in the nonlinear mapping of both structured and unstructured data. These domains include – but are not limited to – image recognition, speech recognition, natural language processing, fraud detection, self-driving cars and robotics. Nonetheless, their applicability to financial markets is still considered a notoriously difficult task. Unlike images, stock markets are determined by human interventions and economic forces, making them endogenous. While a machine can learn to classify cats from dogs, the main challenge in financial markets is that "cats morph into dogs" (Segal 2019). This implies that the training data that investors have access to today are of less relevance in the future. The latter is known as the concept of drift in the ML community.

We investigate the use of machine learning algorithms in the context of market timing strategies aiming at exploiting return predictability, in order to outperform the buy-and-hold strategy. The feasibility of accurately predicting return is discussed in, for example, Campbell and Shiller (1988), Fama and French (1988), Cochrane (2007) and more recently Hull and Qiao (2015) and Jiang *et al*. (2018). Additionally, Faber (2007), Kritzman *et al*. (2012), and Boudt *et al*. (2015), for instance, provide evidence on the alleged benefits of market timing strategies. Kritzman *et al*. (2012) and Boudt *et al*. (2015) show that market timing strategies, using time varying regime probabilities, lead to a reduction of the volatility as well as tempering negative returns of a portfolio strategy. A disadvantage of results obtained using these nonlinear models is that some of them are not straightforward to reproduce.

In response to the above criticism, we use only publicly available market data and open-source software to evaluate the usefulness of several machine learning algorithms for tactical asset allocation between bonds and equities. Under this framework, an investor rotates dynamically between equity and bond funds given the machine learning extracted signal. By its inner construction, the strategy helps manage risk by spreading the investors' wealth between two asset classes on an active basis. Given the proposed strategy, we investigate its potential gains in terms of absolute and risk-adjusted returns, compared to passive funds along with other commonly *ad hoc* techniques.

Since we use only publicly available data and open-source software, the strategy is available to all investors – retail and professional[1]. A very appealing feature of this setup for retail investors resides in the fact that we can use data in an effortless manner to form actionable insights. However, this assumes that retail investors can use advancements in ML and open-source software. A more general target user population would constitute private banks, robo-advisors or institutional investors. Nonetheless, our attempt is to construct a tactical asset allocation rule-based strategy that could potentially help retail investors in their decision-making, whether they trade on their own or delegate their investment to more sophisticated investors. Without loss of generality, we relate our discussion to the retail investor.

Inspired by the framework presented by Hull and Qiao (2015), we implement a tactical asset allocation strategy and conduct our investigation from statistical and economic viewpoints. The presented strategy consists of either taking a position in the SPY ETF or a bond ETF. The SPY ETF tracks a market-cap-weighted index of the stocks listed on the S&P 500, whereas the IEF tracks 7–10 years of Treasury bonds. Unlike Hull and Qiao (2015), our investigation is conducted solely using non-licensed publicly available data. By extracting a data-driven machine learning signal and implementing an extensive backtest between December 2005 and January 2019, we provide evidence that a retail investor can increase their welfare in terms of both raw and risk-adjusted return, compared to holding passive index funds.

Note that the aggregation of the raw extracted signal from ML algorithms into actionable insights is highly dependent on the investors' willingness to take risks, reflected by the threshold probability from where positions will be taken. In the remainder of the chapter we will refer to that threshold as the risk aversion threshold. We find that the investors' willingness to take risks highly influences the financial performance of ML algorithms, in the context of tactical asset allocation. Besides that, the presented implementations resulted in strategies that outperformed a constant mix investment portfolio in terms of out-of-sample return, lower maximum drawdown and lower value at risk, confirming the findings of Faber (2007), Kritzman *et al*. (2012),

1 The implementation is done in the R statistical language (R core team, 2018). All the codes for the analysis can be found at: https://github.com/m-cela/Machine-Learning-Algorithms-in-Tactical-Allocation/blob/master/run_TAA_BCS.R.

and Boudt *et al*. (2015). It is clear from an intuitive perspective that our ability to time the market or correctly predict the future market direction, results in a higher return on the invested wealth. Of all analyzed ML algorithms in this research, the multilayer perceptron (presented in section 2.5) clearly outperforms the investigated ML algorithms both in terms of statistical and financial performance.

The remainder of this chapter is organized as follows. In section 2.2, we describe the analyzed data followed by a section explaining the tactical asset allocation framework. An overview of the investigated machine learning algorithms will then be given in section 2.3 with the corresponding evaluation criteria in section 2.4. Finally, the results of the implemented strategy will be discussed in section 2.5, followed by the shortcomings of research and concluding remarks.

2.2. Empirical investigation

This section describes the data sources and investment problem facing the investor.

2.2.1. *The data*

As in Hull and Qiao (2015), we use several indicators to predict the direction of the future price movement of the S&P 500 ETF. The SPY ETF thus constitutes the central role along the presented analysis[2]. We use the following five different symbols to construct the main feature space:

i) SPY (S&P 500 ETF);

ii) IEF (7–10 year Treasury bond ETF);

iii) VIX index;

iv) GLD (gold ETF);

v) XLF (financial sector ETF).

2 We use the quantmod R package by Ryan and Ulrich (2008) to download ETF price and volume data from Yahoo Finance.

For each asset, we compute the daily return, the daily volume (except for the VIX) and the 25-day return moving average. This constitutes a dataset of 14 features dating between December 27, 2004 and January 31, 2019.

The tactical allocation strategy is to stay invested as long as the S&P 500 ETF is expected to stay flat, or to be rising. Formally, this means that we introduce a dummy variable that is one if the next day return exceeds a threshold, which we calibrate to -1%. All input features will be used to perform a binary classification of the exceedance of the S&P 500 ETF returns with respect to that threshold. Note that we will refer to that threshold as the class assignment threshold.

To understand the intuition behind the feature space, a few comments are in order. First, the SPY and IEF ETFs serve as the two main alternatives facing the investor. While the response variable depends on the direction of the SPY alone, the investment decision involves both ETFs. Additionally, during times of increased uncertainty, investors tend to move their wealth from equity to Treasury bonds, a phenomenon known as *flight-to-quality*. Moreover, since all bonds are subjected to interest rate risk, the inclusion of the IEF provides further information on the interest rate environment in the market. Second, recall that the VIX index represents the expectations of the implied volatility of the S&P 500 index. It is also known as the *fear gauge* that captures investor sentiment in the equity market. Third, we include the gold ETF due to its historical stature in the financial markets as a safeguard of wealth. In addition, it also serves as a proxy for commodity prices in the market. Fourth, the XLF ETF serves as a proxy for economic prosperity, during which banks and other financial institutions expand their balance sheets.

Finally, we standardize all input features to filter out the effect of the scale on which they are measured[3]. We denote the 14 features at day t by x_t. In terms of the response variable, we consider a dummy variable that corresponds to the change in the SPY ETF at day $t+1$. Therefore, the machine learning algorithms perform a binary classification task by mapping

3 For the standardization of the input features, we make use of built-in functionality of the caret package in R (Kuhn 2008). It standardizes the features using a standard z-score transformation such that all features have a zero mean and a variance of one. This is constructed on a rolling basis to avoid any data leakage concerns while allowing a filtering of the scale of each individual input feature.

the input feature space onto the response variable. In other words, the purpose of the ML algorithms is thus to estimate a function at t using the available information denoted by x_t and to construct a predicted (signal) about the change in the SPY ETF at time as $t+1$, denoted by $\hat{y}_{t+1}$. Given the extracted signal, we deploy a tactical asset allocation strategy that rotates between the SPY and the IEF. The following section is dedicated to the discussion of this strategy.

2.2.2. *Tactical asset allocation strategy*

The key idea behind the ML application used in this chapter is to construct a data-driven solution that maps the feature space into a trading signal. Based on the intensity of the signal, the investor takes a corresponding position in the SPY and the IEF. In particular, the weight allocated to the SPY (respectively IEF) is determined by the probability of the equity market going up (respectively down) in the next day given the current information available to the investor.

To put formally, let $\hat{\pi}_{t+1|t} = \hat{\mathbb{P}}\left(r^e_{t+1} > \tau_t \,|\Omega_t\right)$ denote the predicted probability that the equity return exceeds the class assignment threshold τ_t at time $t+1$ given the information available at time t, Ω_t. A natural value for the class assignment threshold τ_t is to set it at zero when there are no transaction costs or estimation error. In practice, we recommend taking a small negative value as -1%, implying that the predicted probability is about the event that the equity market stays flat or rises. The disadvantage of a fixed threshold is that the probability of exceedance also varies when volatility changes. A further improvement may therefore be to take volatility into account when calibrating the value of the class assignment threshold τ_t.

Note that the Ω_t denotes the information set of the feature space dating between the $t-T+1$ and t trading days (included), where T is the sample size used to train the ML algorithm. In short, in our analysis, we will take an observation period T to estimate a mapping function between input features and the response variable of that period. The estimated mapping function will then be used to predict a probability of future price movements $\hat{\pi}_{t+1|t}$. Given the extracted signal, the trading strategy allocates $\hat{\pi}_{t+1|t}$ of his or her wealth to SPY in the following day if and only if $\hat{\pi}_{t+1|t} > a$ for some constant $a \in (0,1)$. Otherwise, he or she allocates 100% of his or her wealth to the IEF.

A couple of points are worth noting. First, the parameter a denotes the level of confidence needed for the investor to allocate part of his or her wealth to the equity market. In other words, it reflects the investors trust in the predicative power of a given ML algorithm. Second, the parameter a also denotes the investor's risk aversion. The larger the a is the more risk averse he or she is. For instance, if the investor is extremely risk averse, i.e. $a \to 1$, then he or she allocates his or her wealth in the Treasury bonds, regardless of the ML signal. In the rest of this chapter, we will refer to the a as the confidence level for short.

2.2.3. *Implementation*

Let r_{t+1}^{e} and r_{t+1}^{b}, respectively, denote the return on equity and bond over a single period (trading day). Also, let $r_{t+1}^{p}(a)$ denote the return of the ML strategy over the next period for an investor with a predetermined confidence level of a, such that

$$r_{t+1}^{p} = \begin{cases} \hat{\pi}_{t+1|t} r_{t+1}^{e} + \left(1 - \hat{\pi}_{t+1|t}\right) r_{t+1}^{b} \; ; \; if \; \hat{\pi}_{t+1|t} \geq a \\ r_{t+1}^{b} \qquad\qquad\qquad\qquad\qquad\quad ; \; if \; \hat{\pi}_{t+1|t} < a \end{cases} \qquad [2.1]$$

Clearly, increasing the value of the risk aversion threshold a results in a strategy that invests more of the wealth in the Treasury bonds and takes a position in the SPY ETF on a less recurring basis. This also indicates a low level of trust by the investor in the ML algorithm. For implementation, we set a to be either 0.5, 0.85, 0.9 or 0.95.

It is obvious from equation [2.1] that the return of the ML strategy depends on the algorithm's ability to forecast the next period movement, denoted by $\hat{\pi}_{t+1|t}$. To compute $\hat{\pi}_{t+1|t}$, we use a window of 250 trading days, i.e. $T = 250$. Hence, the information set at time t, i.e. Ω_t, consists of a 250×14 matrix from which the ML algorithm extracts a mapping function $\widehat{f}_t$. This, as a result, allows the investor to transfer the recent data x_t into a decision rule, i.e. $\widehat{f}_t : x_t \to \hat{\pi}_{t+1|t}$.

We recalibrate the $\widehat{f}_t$ function on a monthly basis, using the recent 250 trading days. To demonstrate this process, suppose that there are D trading days in each month and that t represents the last trading day of the month.

Using all market information between $t - T + 1$ and t (included), we estimate the $\hat{f}_t$ function and back-test the strategy over the next D days, i.e. between $t + 1$ and $t + D$ (included). After observing the recent D days, we update the estimation window to include the recent data and to estimate the mapping function at $t + D$. Using the updated function, $\hat{f}_{t+D}$, we back-test the strategy over the next month, i.e. between $t + \mathrm{D} + 1$ and $t + 2D$ (included). We repeat this process until the end of the sample. Recall that the full sample period dates between December 27, 2004 and January 31, 2019. Given an initial sample of 250 days, this corresponds to 158 mapping functions for each machine learning algorithm.

Note that estimating the mapping function requires estimating the hyperparameters of the ML algorithm. To do so, we conduct a 10-fold cross validation with 3 repeats and 45 different fixed seeds at each month, which results in 45 extracted signals for the same feature. Finally, we average the signal among the 45 seeds to form the tradable signal.

2.2.4. *Benchmarks*

For comparison, we consider two common benchmarks. The first one is a passive 60-40 benchmark (naive) that allocates 60% and 40% to the SPY and IEF, respectively. We refer to this benchmark as naive since it does not incorporate any information in the allocation strategy. Additionally, according to the literature on the portfolio optimization under parameter uncertainty (estimation risk), it is well established that the naive strategy constitutes a difficult benchmark to outperform – see for example, DeMiguel *et al*. (2007). For the other benchmark, we consider a similar strategy to the one proposed by Faber (2007). In particular, we hold a 100% position in the SPY ETF at time $t + 1$, if the price at t exceeds the corresponding 200-day moving average. Otherwise, the position holds the 100% in the IEF ETF. This is a slight modification from Faber's original benchmark, which holds cash rather Treasury bonds.

2.3. A review of machine learning algorithms for prediction of market direction

This section contains a brief overview of several machine learning algorithms implemented in our empirical investigation. Additionally, at the

end of the section, we describe the tuning process implemented to estimate the hyperparameters needed for each algorithm[4].

2.3.1. *K-nearest neighbors*

We start our review with a simple algorithm known as the K-nearest neighbor (KNN) classifier. The idea behind the KNN algorithm is to make use of the concept of similarity between an event in the test set and its K-nearest neighbors in the training set. For our purpose, given observation x_t, we need to determine whether it is associated with an increase/decrease in the SPY over the next day. If $K = 1$, then the algorithm returns $\hat{\pi}_{t+1|t} = 1$, if the nearest neighbor in the training set of the recent T observations were associated with a positive return in the SPY. Alternatively, the algorithm would return $\hat{\pi}_{t+1|t} = 0$, if the neighboring observation were associated with an equity market downturn. Clearly, under $K = 1$, the predicted probability $\hat{\pi}_{t+1|t}$ is either 1 or 0. However, by considering a larger number of neighbors (larger K), the probability is determined by the number of neighboring observations in which the SPY market went up.

To determine whether two observations x_a and x_b are neighbors, we need to compute the Euclidean distance between their feature spaces:

$$d_{ab} = \left(\sum_{j=1}^{z}\left(x_{aj} - x_{bj}\right)^2\right)^{1/2} \quad [2.2]$$

with z denoting the number of features, i.e. 14. For instance, if $K = 10$, then we need to find the 10 observations out of T in the training set that have the closest Euclidean to the observation. Each of the 10 observations has a label indicating 1 or 0 if the SPY went up or down (i.e. whether the return exceeded the class assignment threshold τ_t), respectively. Given these labels, we can determine the probability $\hat{\pi}_{t+1|t}$. For example, if we find that 7 out of these 10 neighbors were associated with an increase in the SPY over the next day, then the algorithm determines that $\hat{\pi}_{t+1|t} = 0.7$.

4 The implementations of the machine learning algorithms are tested using the caret package (Kuhn 2008). This package has the advantage of being compatible with over 200 other libraries, allowing for a more integrated use in terms of pre-processing, as well as hyperparameter selection. The complete overview of all compatible packages can be found at http://topepo.github.io/caret/available-models.html. Several potential hyperparameters proposed by the caret package are used iteratively in a repeated cross validation setup to determine the optimal one from the training data. The standard configuration was used for all investigated ML algorithms to avoid overfitting of the hyperparameters on the training period.

Note that by applying a classification based on a distance metric, the classifier becomes sensitive to the scale of the input features and by performing standardization on all input features enables each feature to contribute equally to the prediction (Mohamad and Usman 2013). For a more detailed description of the algorithm we refer to Kuhn and Johnson (2013). It is related to scenario generation based on Mahalanobis distances between past and target values for state variables, as explained in Meucci (2012).

2.3.2. *Generalized linear model*

The workhorse model to predict a dummy variable is the generalized linear model (GLM), proposed by Nelder and Wedderburn (1972). In the context of a logistic regression for a binary classification, the probability of classifying x_t following equation into a positive event is given by:

$$\mathbb{P}\left(r_{t+1}^{e} \geq \tau_t \mid x_t\right) = \frac{1}{1+e^{-\left(\beta_0 + x_t^T \beta\right)}} \quad [2.3]$$

Given a training sample of T observations, the coefficients β_0 and β can be estimated using a maximum likelihood estimation (MLE) approach. Nonetheless, note that a logistic regression does not perform any cross validation to estimate these coefficients. Hence, it does not impose any regularization and is subjected to a risk of overfitting.

2.3.3. *Elastic net regression*

A possible remedy for the overfitting problem associated with the logistic regression is to add a penalty to the MLE procedure. One common approach is known as the elastic net for GLM models, proposed by Freedman *et al.* (2010). It aims to maximize the likelihood function of the training data while imposing a penalty on the first and second norms of the coefficients β_0 and β. In other words, it aims to solve the following optimization problem:

$$max_{(\beta_0,\beta)\in\mathbb{R}^{z+1}} \left[\frac{1}{T}\sum_{i=1}^{T} I\left(r_{t+1}^{e} \geq \tau_t\right) log p(x_i) + I\left(r_{t+1}^{e} < \tau_t\right)\ log\left(1 - p(x_i)\right) - \lambda P_\alpha(\beta)\right] \quad [2.4]$$

$$P_\alpha(\beta) = \sum_{j=1}^{z}\left[\frac{1}{2}(1-\alpha)\beta_j^2 + \alpha\left|\beta_j\right|\right] \quad [2.5]$$

with z denoting the number of features, λ the magnitude of the penalty, α the elastic net hyperparameter and $p(x_i)$ is short for $\mathbb{P}(r_{i+1}^e \geq \tau_t \mid x_i)$, i.e. the probability of an upward movement in the SPY ETF at time i+1, given the information available at time i or more formally $\mathbb{P}(r_{i+1}^e \geq \tau_t \mid x_i)$.

The chosen level of α in the elastic net regression determines the relative importance of applied penalty. Hence, in the extreme case of α = 1, the applied regression becomes a lasso regression as in the case of $\alpha = 0$ the applied regression becomes a ridge regression. Both types of penalties aim to add bias in the predictions during the training phase to improve the generalization on the unseen data. From a broad perspective, both types of regularization work in a similar fashion, but differ in terms of the final output. In particular, the ridge regression, on the one hand, tends to shrink correlated predictors towards each other. The lasso regression, on the other hand, tends to give a higher relative importance to sub-features and does serve as an elimination process to determine the more predictive features.

2.3.4. *Linear discriminant analysis*

Introduced by Fisher (1936), the linear discriminant analysis (LDA) used in a binary classification setup starts from a basic intuition of searching a linear combination of variables that best separate the two classes. This is done by optimizing the parameters of the linear combination such that the ratio of between class variance (or centered mean of classes) to the within class variance is maximized. The latter assumes that the input features follow a multivariate Gaussian distribution with class specific vectors and a common covariance matrix. By applying the linear combination of the input features, the algorithm in fact applies a dimension reduction with the highest separability between classes.

For more details on the inner working of this algorithm, we refer to Chapter 4 of the book of Fukunaga (2013)[5].

2.3.5. *Support vector machines with radial kernel*

Support vector machines (SVM) are based on the idea of class separation applied by finding an optimal separating hyperplane that maximizes the

5 We use the MASS library by Venables and Ripley (2002) to implement the linear discriminant analysis in this research.

margin between the class's closest points. It thus takes only small proportion of points, called the support vectors, that lies on the boundaries into consideration. Note that by only using the support vectors to determine the optimal boundaries, the procedure results in an efficient machine learning algorithm. Put formally, the support vector machine with radial basis can be expressed by the following Lagrangian dual:

$$min_{\alpha}\ 1/2\ \sum_{i=1}^{m}\sum_{j=1}^{m} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{m} \alpha_i \qquad [2.6]$$

subject to $\sum_{i=1}^{m} y_i \alpha_i = 0$ and $D \geq \alpha_i \geq 0$ for $i = 1, \ldots, m$, with $K(x, y) = exp(-\sigma \|x - y\|^2)$ and σ a hyperparameter tuning the influence of individual instances to the decision boundary.

Note that the kernel function $K(x, y)$ allows the algorithm to form nonlinear hyperplanes and that the upper bound D (also called cost) forms the soft margin for the linear inseparable cases. Placing the value of D too high could lead to overfitting on the training data while setting it too low could result in overgeneralization. Figure 2.1 illustrates the effect of the hyperparameters and individual components of the support vector machine with radial kernel function, on a dataset that is used as a toy example.

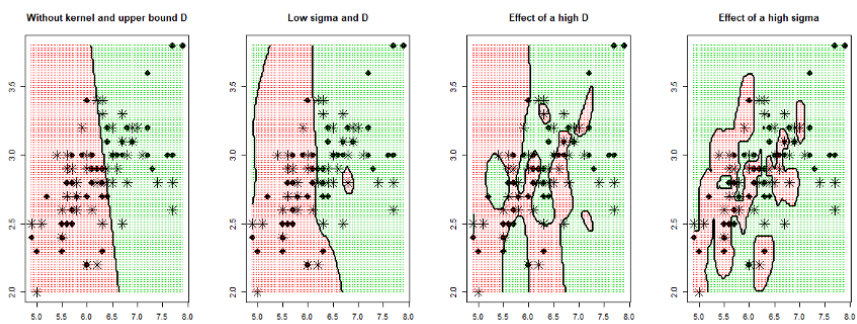


Figure 2.1. *Effect of hyperparameters and kernel function on decision boundaries of a support vector machine*

It can be noted from Figure 2.1 that the use of radial kernel function results in nonlinear decision boundaries. Furthermore, by comparing the two central plots, we can see the effect of a high upper bound D on the decision

boundary of the classifier. Setting a high upper bound on the classifier leads to an increased fitting of the decision boundaries on instances such that only a small proportion is wrongly classified. At last, an increase in the hyper parameter sigma causes individual instances to highly influence the decision boundary. The latter results in a localized decision boundary around groups of instances from the same class. For more details on the explanation of the support vector machine with a radial kernel function algorithm, we refer to Cortes and Vapnik (1995)[6].

2.3.6. *C5.0*

The C5.0 algorithm introduced by Quinlan (1993) is a classification tree with a splitting criterion based on the notion of entropy which is a measure of uncertainty from information theory (Shannon, 1948). Classification trees consist of a series of "if then" statements that can be visualized in a tree structure. Each node of the tree splits the predictors into smaller subsets with a tree containing a higher concentration of a given class. The splitting procedure consists first of selecting a feature that contains the most information and estimating an optimal split value that maximizes the information gain. In a binary classification setup with p being the probability of appurtenance to the first class, the information statistic is given by:

$$-[plog_2 p + (1-p)log_2(1-p)] \qquad [2.7]$$

Note that p is derived from the division of the sum of the number of instances corresponding to the first class before and after the split to the total amount of instances analyzed.

We can then derive the information gain from the information statistic by taking the difference in information prior to the split to the information after split. This tree generating process then leads to a fully formed tree that can perform the binary classification task. In order to form the final decision tree used to make predictions, the C5.0 algorithm performs a final pruning procedure that consists of methodically removing branches that do not contribute enough to general classification. Sub-trees are removed until the error rate exceeds the one without pruning, leading to the final classification tree.

6 The implementation of the support vector machine with a radial kernel function in this research was performed through the R package kernlab (Karatzoglou *et al*. 2004).

2.3.7. *Random forests*

The random forest (RF) algorithm was first introduced by Breinman (2001) and builds on the notion of classification trees, as described in the C5.0 section of the book of Kuhn and Johnson (2013). The algorithm uses a bootstrapped dataset which consists of randomly selecting samples of the original dataset. Note that the bootstrap method allows the same entity to be picked more than once and that typically about one third of the dataset is left over (called the out-of-bag samples). Decision trees are then created using the bootstrapped dataset by only using a subset of the features, at each step. In order to reduce the correlation among predictors, randomness was introduced in the tree construction process which randomly splits the features.

Aggregating the results of the bootstrapped dataset is called bagging and forms the final output of a random forest algorithm. Thus, the output probability, as presented in this research, is derived from the proportional occurrence of a particular class from all formed trees. The overall accuracy of a random forest model can be measured by the proportion of correctly classified out-of-bag samples.

2.3.8. *Multilayer perceptron*

A multilayer perceptron (MLP) was first introduced by Rosenblatt (1958) in its single layered configuration and later generalized to its multilayered configuration by Rumelhart *et al*. (1985). This algorithm consists of a chosen number of consecutive layers of nodes that perform a linear transformation of the feature space with some bias term. On top of the performed linear transformation, there is an additional nonlinear transformation applied – known as the activation function. More formally, the activation function applied to the linear transformation of the input on a single node can be described in the following equation:

$$Z_m = \sigma(x_t' \alpha_m + b_m), \forall m = 1, \dots M \qquad [2.8]$$

where Z_m is the value of the feature space x_t mapped into the first hidden layer of node m, σ is the activation function, α_m is the vector of weights that maps the feature space into node m and b_m is the bias term. We use the typical sigmoid function as the activation function, which is given by $\sigma(k) = \frac{1}{1+e^{-k}}$ for a given scalar k. A caveat is that the applied activation function may be subject to a vanishing gradient descent.

If there is a single layer, then this resembles a plain vanilla neural network with a single hidden layer. In such a scenario, there is an additional transformation that maps the vector of the hidden nodes $Z \in \mathbb{R}^M$ into the event of SPY going up or down in the next period. In this case, it follows that the probability of the market going up in the following period is given by the consecutive transformation of the input features, through equation [2.7].

If there were multiple layers, then we need to consider additional mappings. However, the same reasoning applies. In order to establish a forecast for the SPY going up, i.e. $\hat{\pi}_{t+1|t}$, we need to estimate the corresponding parameters, which is achieved using the back-propagation approach. This approach makes use of a loss function that is consecutively minimized by adapting the relevant weights and biases, in order to increase the accuracy of the neural network.

2.3.9. *Model averaging*

As an additional prediction, we consider a weighted version of all algorithms reviewed above. In other words, each model produces its own $\hat{\pi}_{t+1|t}$. Hence, given all of which, we provide a weighted score where the weighting scheme is proportional to the inverse of the in-sample mean squared error of each model[7]. Such an approach is inspired by the research of Stock and Watson (2004). Put formally, the weight assigned to algorithm a is given by:

$$w_{at} = {SSE_{at}^{-1}}\Big/{\sum_{a=1}^{A} SSE_{at}^{-1}} \qquad [2.9]$$

with

$$SSE_{at} = \sum_{s=t-T}^{t-1}\left(\pi_{s+1}^{a} - \hat{\pi}_{s+1|s}^{a}\right)^2 \qquad [2.10]$$

denoting the sum squared error score derived the difference between the actual output and the predicted output for period t and algorithm a, for all

7 The in-sample mean squared error is computed using a rolling evaluation approach that only uses the information available up to time t in order to avoid any look-ahead bias in the strategy.

$a = 1, .., A$. Let $\bar{\pi}^{S}_{t+1|t}$ denote the weighted forecast using the A algorithms with respect to the sum-squared errors, such that:

$$\bar{\pi}^{S}_{t+1|t} = \sum_{a=1}^{A} w_{at} \, \hat{\pi}^{a}_{t+1|t} \quad [2.11]$$

Note that w_{at} and $\hat{\pi}^{a}_{t+1|t}$ are calculated on a monthly and daily basis, respectively. As a result, the value of w_{at} is kept constant over the prediction period until the next training of the algorithm. This will further be denoted as "weighted" in the remainder of this chapter. As an additional measure, we consider an equally weighted average output, which is given by:

$$\bar{\pi}^{e}_{t+1|t} = {}^{1}/_{A} \sum_{a=1}^{n} \hat{\pi}^{a}_{t+1|t} \quad [2.12]$$

Note that, overall, we have $A = 8$ algorithms used in our study. We will refer to that output as "Average" in the remainder of this chapter.

2.3.10. *Repeated k-fold cross validation*

For most algorithms deployed in this research, we need to specify the hyperparameters. For instance, in an elastic net GLM, we need to determine the λ. To determine these parameters in a data-driven way, we deploy a repeated k-fold cross validation. The k-fold cross validation method consists of partitioning the training data into k mutually exclusive, randomly formed sub-sets of approximately equal sizes. The procedure works by iteratively holding one-fold out of the remaining training data that is fed into a machine learning algorithm. The performance of the algorithms is then derived through averaging the accuracy of the prediction made on the fold held out of the training set.

In this research, we make use of the repeated k-fold cross validation. This version of the k-fold cross validation repeats the k-fold cross validation method a given amount of times and averages the results. Typically, the 10-fold cross validation with three repeats was applied to determine the hyperparameters of all tested ML algorithms. The accuracy of a model is thus derived by the average performance of 30 held out samples. Note that on top of the presented repeated k-fold cross validation, the output of each individual ML algorithm will be tested on 45 different fixed seeds and averaged to form the final output signal.

2.4. Evaluation criteria

We evaluate performance from a statistical and economic perspective. The first consists of conventional statistical tools such as the accuracy of prediction, kappa statistic, area under the curve (AUC), specificity and sensitivity. The aim behind the use of those metrics is to make comparisons across models possible. Note that the cost attached to the decisions is not included in this part of the metrics. In other terms, the financial consequence attached to a classification is not taken into account for those metrics. From a practical perspective, these metrics are built-in functions from the caret package on the exception of area under the curve that uses the glmnet package.

The economic metrics include the annualized returns, Sharpe ratio, skewness, kurtosis, maximum drawdown and historical value at risk (VaR) with a confidence level of 95%. These metrics are computed using the PerformanceAnalytics package (Peterson and Carl 2018).

Note that both the statistical and economic performance of the algorithms are reported for the out-of-sample period, which ranges from December 2, 2005 until January 31, 2019.

2.4.1. *Statistical performance*

Selecting the right performance metrics for the used forecasting models in this research is essential to evaluate how suited the ML algorithms are for the investment problem. In this optic, the overall accuracy of a model can be derived as the correctly identified classes divided by the total amount of instances, given a specific cutoff probability. Similarly, the sensitivity and specificity can be derived from the following equations (Bradley 1997):

$$\text{Sensitivity} = \frac{TP}{FN + TP} \; ; \; \text{Specificity} = \frac{TN}{FP + TN} \qquad [2.13]$$

With TP and TN denote, respectively, the true positives and negatives, while FP and FN denote, correspondingly, the false positives and negatives.

Put intuitively, sensitivity in our case denotes the proportion of time the ML algorithm is capable of identifying the SPY going up in the following

period out of all periods in which it did. A small sensitivity value implies a large opportunity cost for the investor. On the other hand, specificity denotes the proportion of time that the ML algorithm fails to identify a negative event (SPY drops the next day). A small value of specificity implies greater exposure to downside risk than expected.

To reconcile between the two, it is common to plot the sensitivity and specificity with a varying cutoff probability. This forms the receiver operating characteristic (ROC) curve from which the AUC can be derived using a trapezoidal integration. The AUC is a single scalar that represents the probability that a randomly chosen positive example is correctly assigned, with a greater probability than a randomly chosen negative example (Ben-David 2008). An AUC value of 1 implies that the ML algorithm is capable of identifying all positive and negative events separately. On the other hand, if the algorithm is incapable of distinguishing between positive and negative events, we expect an AUC of 0.5. Moreover, an AUC value of 0 indicates that the ML algorithm is predicting positive events as negative events and negative events as positive events. For our analysis, the closer the AUC value is to 1, the better the statistical performance of the algorithm.

In the case of a strong class imbalance, we should be careful with the result derived from the accuracy. For example, let us suppose the case where 95% of the target variables are assigned to a particular class. In this example, a model that would assign all the input to this given class would result in a classifier with an accuracy of 95%, without reflecting the desired classification ability of a model.

To adjust for the problem of assignment by chance, we use Cohen's kappa (Cohen 1960). This is a metric that represents the accuracy corrected to the assignment to a particular class due to chance. More formally, Cohen's kappa is calculated with the following equation:

$$Kappa = \frac{p_0 - p_c}{1 - p_c} \quad [2.14]$$

with p_0 denoting the accuracy and p_c the agreement probability due to chance, which is defined as follows:

$$p_c = \sum_{j=1}^{2} p_{j.}\, p_{.j} \quad [2.15]$$

where $p_{j.}\, p_{.j}$ denote rows and column marginal probabilities.

2.4.2. *Financial performance*

We consider several metrics to measure the financial performance of the investment algorithms based on the predictions. Given the time series of the daily portfolio return from equation [2.16], we compute the annualized return (AR) as:

$$AR = \left(\prod_{t=1}^{n}\left(1 + r_t^p\right)\right)^{252/n} - 1 \qquad [2.16]$$

where AR denotes the annualized return, r_t^p the daily returns from equation [2.17] at time t and n the observed period which is equal to 3,322 analyzed days for this research.

As a risk-adjusted return, we calculate the annualized Sharpe ratio (Bacon, 2008) as:

$$S = \frac{AR}{\sqrt{252}\, s} \qquad [2.17]$$

with

$$s = \sqrt{\frac{\sum_{t=1}^{n}\left(r_t^p - \bar{r}\right)^2}{n}} \qquad [2.18]$$

The maximum drawdown (MDD) represents the maximum loss an investor could have made during the investment period. This value is derived as the difference at the highest value compared to the lowest value of the cumulated returns of a strategy. In addition, the historical VaR representing the negative value of the 5% quantile of the portfolio returns will be analyzed. Furthermore, the turnover and equivalent transaction costs will be incorporated in the analysis to identify the usefulness of the strategy from a more practical perspective. Both metrics are commonly used in the transaction cost literature (see, for example, DeMiguel *et al.*, 2013). Note that the turnover simply represents the switches in the taken positions and the equivalent transaction cost represents a hypothetical measure at which an ML-based portfolio return, equals the returns of a benchmark with transaction cost included. It thus represents the maximum transaction cost that is allowed by the strategy to safeguard the same annualized returns as the best-performing benchmark. At last the turnover will be annualized by dividing it by the amount of years.

2.4.3. *Significant features*

The support vector machines, the C5.0, random forests and multilayer perceptron are highly nonlinear ML algorithms. The nonlinearity is an advantage in terms of flexibility, and also leads to opaqueness in terms of the underlying reasons behind their decision-making. A solution for this is to make use of the LIME framework, originally developed by Ribeiro *et al*. (2016) to uncover the important features from the "black box".

LIME stands for Local Interpretable Model-agnostic Explanation. It encompasses a framework that describes a given ML algorithm locally, through a linear approximation. In the context of a binary classification, the approximation is done through a logistic regression with a lasso regularization. The local description points to the fact that each instance is analyzed individually and that variations of the analyzed instance are used to make the local approximation. This is done by adding noise to the features to investigate its effect on predictions. Note that the LIME framework has the advantage of being a model-agnostic that provides an understandable explanation independent of the deployed ML algorithm[8].

The output of this implementation illustrates the effect or contribution of each feature to the prediction of an analyzed instance. Similarly, the framework presents how changes in an input feature impact a change in the predictions. For the purpose of our investigation, we analyze the best-performing model, in which we keep 90% of the dataset for training and the rest for sensitivity testing. Each individual prediction is subject to the linear local approximation that, eventually, is averaged and will be presented in section 2.5. We then use the averaged coefficients of the logistic regression with a lasso regularization, along with the average value of input features, to determine the alleged effect of each individual feature on an average basis. For a more detailed view on the framework, refer to Ribeiro *et al*. (2016) or Molnar (2018).

2.5. Results and findings

In this section, we discuss our main results and findings. We set our discussion with basic summary statistics, followed by a summary of the statistical and financial performance of the ML-based investment strategies.

8 From a practical perspective, the LIME framework is implemented in R through the iml package of Molnar *et al*. (2018).

We finalize the section with an investigation of the important features from the best-performing ML algorithm.

2.5.1. *Descriptive statistics of the data*

As an initial perspective on the data, we plot the feature space in Figure 2.2 over the whole sample. Figure 2.2 illustrates the distribution of each individual feature using a boxplot. Note that for each feature, the values are standardized such that each has a zero mean and a unit variance. There is a clear heterogeneity in the data and also evidence of non-normality in the features.

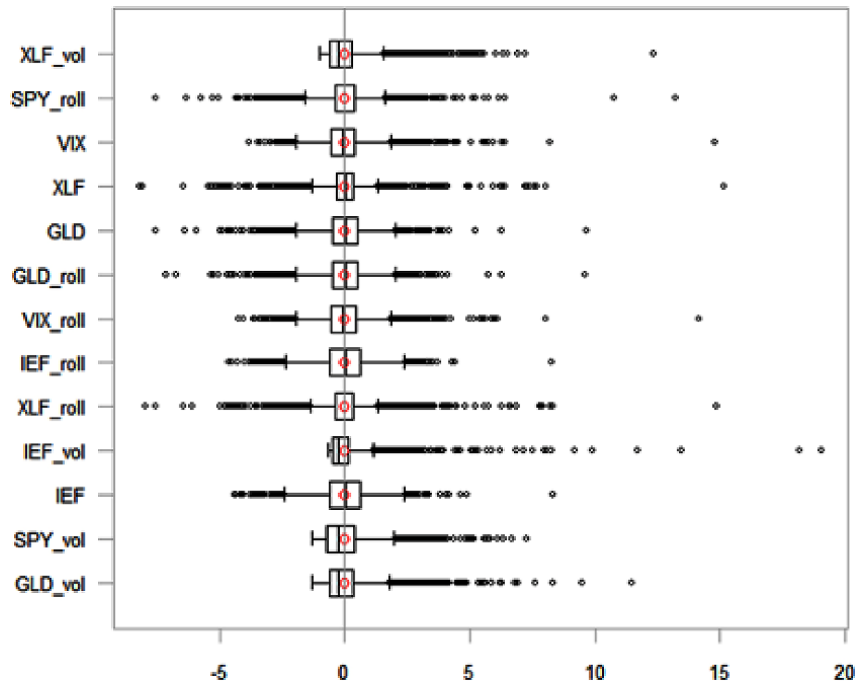


Figure 2.2. *Boxplot of the standardized input features. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

In Table 2.1, we report the correlation matrix among the features. We observe that there is a strong correlation between the SPY and the IEF, VIX, XLF, along with their MAs. While we approximate economic conditions using the XLF ETF, we note that a significant proportion of the constituents

of the SPY comes from the financial sector. We observe a weaker negative correlation between the SPY ETF and the VIX index. In terms of signs, we observe a negative correlation between the SPY and the IEF. The same holds true for the VIX index. As conjectured, IEF denotes a haven for investors during times of uncertainty, whereas the VIX captures the increased uncertainty in the equity market.

	SPY	VIX	GLD	IEF	XLF	SPY_roll	VIX_roll	GLD_roll	IEF_roll	XLF_roll	SPY_vol	GLD_vol	IEF_vol	XLF_vol
SPY	1	-0.73	0.031	-0.408	0.821	0.987	-0.717	0.037	-0.408	0.811	-0.129	-0.075	-0.023	-0.051
VIX	-0.802	1	-0.03	0.335	-0.554	-0.724	0.989	-0.034	0.336	-0.55	0.135	0.085	0.035	0.065
GLD	0.049	-0.028	1	0.152	-0.045	0.031	-0.028	0.982	0.145	-0.04	0.011	-0.073	-0.002	0.013
IEF	-0.372	0.313	0.151	1	-0.368	-0.402	0.33	0.148	0.981	-0.361	0.088	0.026	-0.008	0.016
XLF	0.843	-0.673	-0.025	-0.386	1	0.812	-0.546	-0.042	-0.365	0.987	-0.084	-0.038	-0.015	0.02
SPY_roll	0.980	-0.802	0.045	-0.371	0.827	1	-0.729	0.034	-0.41	0.822	-0.056	-0.044	-0.019	0
VIX_roll	-0.779	0.984	-0.025	0.309	-0.657	-0.803	1	-0.03	0.337	-0.554	0.082	0.059	0.024	0.036
GLD_roll	0.055	-0.032	0.973	0.148	-0.02	0.052	-0.028	1	0.149	-0.042	-0.011	-0.104	0.003	-0.005
IEF_roll	-0.369	0.318	0.147	0.98	-0.378	-0.376	0.318	0.153	1	-0.367	0.037	-0.001	-0.008	-0.013
XLF_roll	0.829	-0.674	-0.021	-0.383	0.975	0.841	-0.675	-0.021	-0.387	1	-0.022	-0.011	-0.012	0.056
SPY_vol	-0.083	0.035	0.004	0.073	-0.096	-0.054	0.003	-0.02	0.031	-0.061	1	0.589	-0.066	0.838
GLD_vol	-0.038	0.005	-0.007	0.007	-0.037	-0.036	-0.009	-0.041	-0.013	-0.034	0.646	1	0.007	0.589
IEF_vol	-0.005	-0.01	-0.037	-0.02	0.014	-0.007	-0.014	-0.017	-0.013	0.006	0.012	0.196	1	-0.035
XLF_vol	-0.035	0.002	-0.001	0.027	-0.034	-0.017	-0.016	-0.02	-0.001	-0.02	0.816	0.696	0.208	1

Table 2.1. *Correlation matrix of standardized input features with the Pearson correlation on the upper triangle and the Spearman correlation on the lower triangle of the matrix*

The target variable is the dummy variable indicating that the equity return exceeds the class assignment threshold (i.e. $r^e_{t+1} > \tau_t$). We set the return threshold τ_t to –1%. It follows that the analyzed period is characterized by a strong class imbalance with a majority contained in an upward movement over the analyzed period. In fact, we find that the downward movement represents 11.97% of the predicted target classes. In the decision rule, the class imbalance is compensated by setting the probability classification threshold value of the class association a, at a higher level. In this research we will investigate threshold values of 0.5, 0.85, 0.9 and 0.95.

2.5.2. *Statistical performance*

We summarize the statistical metrics in Table 2.2 for all ML algorithms considered in our analysis. The results are summarized with respect to different thresholds a. We note that when $a = 0.5$ all ML algorithms provide a higher accuracy, except for the C5.0. However, accuracy by itself is insufficient to determine the statistical performance. For instance, if the

true data consist of positives 90% of the time, then an algorithm that predicts positive outcomes all the time would exhibit a 90% accuracy. Nonetheless, from a risk management perspective, such an algorithm would fail to predict any negative events. In fact, when we consider a more conservative a level, the algorithm would predict fewer positives. This is manifested by the decrease in the accuracy levels among all ML algorithms.

a=0.5	RF	C5.0	MLP	KNN	GLM	Glmnet	LDA	SVM	Weighted	Average
Accuracy	0.873	0.405	0.875	0.874	0.864	0.874	0.861	0.878	0.875	**0.877**
Kappa	0.048	-0.015	0.041	0.034	0.084	0.042	**0.087**	0.015	0.021	0.026
Sensitivity	0.988	0.381	0.993	0.991	0.972	0.991	0.968	**0.999**	0.995	0.996
Specificity	0.042	**0.576**	0.032	0.03	0.086	0.034	0.094	0.01	0.017	0.020
AUC	0.607	0.49	0.631	0.598	0.593	0.615	0.587	**0.689**	0.606	0.640
a=0.85										
Accuracy	0.655	0.122	**0.722**	0.705	0.695	0.718	0.707	0.681	0.661	0.572
Kappa	0.126	0	**0.159**	0.1	0.095	0.148	0.108	0.09	0.144	0.104
Sensitivity	0.668	0	**0.755**	0.747	0.734	0.752	0.747	0.716	0.671	0.558
Specificity	0.564	**1**	0.49	0.406	0.416	0.475	0.419	0.431	0.594	0.672
AUC	0.554	NA	**0.566**	0.541	0.539	0.561	0.545	0.537	0.562	0.550
a=0.9										
Accuracy	0.533	0.122	**0.643**	0.47	0.616	0.57	0.627	0.537	0.512	0.317
Kappa	0.087	0	**0.14**	0.047	0.108	0.109	0.109	0.086	0.092	0.038
Sensitivity	0.509	0	**0.645**	0.441	0.618	0.553	0.634	0.516	0.478	0.238
Specificity	0.7	**1**	0.626	0.682	0.599	0.692	0.579	0.687	0.752	0.887
AUC	0.545	NA	0.561	0.527	0.548	0.553	0.548	0.544	0.55	0.539
a=0.95										
Accuracy	0.373	0.122	0.423	0.459	0.454	0.321	**0.47**	0.244	0.301	0.122
Kappa	0.049	0	0.061	0.044	0.063	0.035	**0.068**	0.02	0.034	0.001
Sensitivity	0.307	0	0.369	**0.427**	0.411	0.245	0.431	0.15	0.217	0
Specificity	0.845	**1**	0.805	0.69	0.761	0.869	0.749	0.921	0.899	0.998
AUC	0.54	NA	0.541	0.526	0.539	0.535	0.54	0.531	0.538	**0.689**

Table 2.2. *Out-of-sample statistical performance of the strategy with different risk aversion thresholds α. The best performance is indicated in bold for each criterion*

A similar logic applies to the sensitivity and specificity metrics. The former resembles a bullish measure, i.e. higher sensitivity indicates a more positive outlook by the algorithm. The latter is the more bearish measure, i.e. higher specificity implies more negative prediction. As demonstrated in Table 2.2, we observe that sensitivity (respectively specificity) is highest

(respectively lowest) for the $a = 0.5$ level, whereas it decreases (respectively increases) when a increases.

Note further that the kappa statistics tend to have a more mitigated result by displaying an increase in value, along with an increase in the risk aversion threshold, until the threshold exceeds the value of 0.9.

Looking at the average of the ML predictions, we observe that the AUC measure is relatively low (around 0.5) in most cases as compared to ML applications in other fields. However, there are a number of cases in which the metric exhibits larger values. Those are mainly evident when $a = 0.5$. Nonetheless, when we increase the a level, we are intervening with the prediction model by identifying positive events as negative events. For this reason, we observe a decrease in the AUC values. Nonetheless, from a risk management perspective, investors could be more concerned with the downside risk (specificity) than with the opportunity cost of the equity market going up (sensitivity). To further understand the statistical implications of these, we turn to the financial performance evaluation.

2.5.3. *Financial performance*

The following results illustrate the financial performance of the tactical asset allocation strategy. Recall that the implementation consists of one-year training data to predict future stock movement over the coming month. By applying this methodology on rolling window, we obtain 158 recalibrations for the out-of-sample period starting on December 2, 2005 until January 31, 2019.

Table 2.3 illustrates the out-of-sample financial performance for all analyzed ML algorithms. The difference between sections 2.5.2 and 2.5.3 lies in the fact that the magnitude or consequence of the classification is incorporated in the performance in terms of investment. The latter also constitutes the most important factor for the application of an ML algorithm used in the context of an algorithmic trading strategy.

a=0.5	RF	C5.0	MLP	KNN	GLM	Glmnet	LDA	SVM	Weighted	Average
Annualized Return	0.042	0.071	**0.1**	0.075	0.089	0.08	**0.1**	0.078	0.055	0.069
Annualized Std Dev	0.134	**0.091**	0.145	0.153	0.146	0.141	0.145	0.15	0.135	0.133
Annualized Sharpe	0.31	**0.776**	0.692	0.493	0.612	0.565	0.688	0.521	0.407	0.52
Skewness	-0.613	0.531	-0.482	0.166	-1.045	-0.861	-1.05	0.079	-0.852	-0.874
Excess Kurtosis	6.962	19.768	10.704	17.329	10.309	7.817	10.582	11.521	6.461	7.715
Maximum Drawdown	0.609	**0.259**	0.543	0.474	0.559	0.571	0.558	0.458	0.597	0.544
Historical VaR (95%)	-0.014	**-0.008**	-0.014	-0.015	-0.014	-0.014	-0.014	-0.015	-0.014	-0.013
Annualized Turnover	6.714	3.786	3.143	4.429	9.286	3.857	10.286	**0.857**	2.714	2.571
Maximum cost (%)	-0,977	-0.911	-0.208	-0.758	-0.163	-0.727	**-0.05**	NA	-2.177	-1.834
a=0.85										
Annualized Return	0.129	0.046	**0.252**	0.085	0.087	0.13	0.105	0.077	0.152	0.163
Annualized Std Dev	0.099	**0.066**	0.115	0.13	0.128	0.114	0.126	0.115	0.097	0.089
Annualized Sharpe	1.296	0.704	**2.186**	0.657	0.678	1.139	0.831	0.667	1.569	1.839
Skewness	-0.473	0.116	-1228	0.218	-1.478	-1.385	-1458	-0.442	-0.438	-0.426
Excess Kurtosis	2.556	2.648	14.562	25.934	14.982	14.744	15.465	5.283	2.738	3.33
Maximum Drawdown	0.381	**0.104**	0.494	0.414	0.566	0.39	0.486	0.2	0.23	0.148
Historical VaR (95%)	-0.01	**-0.007**	-0.01	-0.012	-0.012	-0.011	-0.011	-0.012	-0.01	-0.008
Annualized Turnover	64.214	**0**	50.071	68.357	43.071	39.5	41.071	14.643	50.643	47.786
Maximum cost (%)	0.023	**4.823**	0.239	-0.022	-0.029	0.075	0.01	-0.136	0.086	0.112
a=0.9										
Annualized Return	0.153	0.046	**0.287**	0.086	0.117	0.159	0.105	0.112	0.168	0.127
Annualized Std Dev	0.088	**0.066**	0.103	0.099	0.114	0.096	0.113	0.096	0.084	0.074
Annualized Sharpe	1.739	0.704	**2.822**	0.87	1.021	1.657	0.935	1.156	1.988	1.717
Skewness	-0.432	0.116	-0.99	-0.904	-1.412	-0.84	-1.361	-0.406	-0.309	-0.126
Excess Kurtosis	3.528	2.648	15.541	8.632	16.027	9.144	16.192	4.214	3.302	3.208
Maximum Drawdown	0.207	**0.104**	0.274	0.288	0.35	0.252	0.338	0.104	0.168	0.123
Historical VaR (95%)	-0.009	-0.007	-0.009	-0.009	-0.011	-0.009	-0.01	-0.009	-0.008	-0.007
Annualized Turnover	69.929	**0**	56.643	86.5	44.5	42.643	44.357	20.786	53.643	37.5
Maximum cost (%)	0.062	**4.823**	0.251	-0.009	0.028	0.129	0.006	0.043	0.102	0.083
a=0.95										
Annualized Return	0.129	0.046	**0.206**	0.087	0.108	0.117	0.133	0.059	0.112	0.046
Annualized Std Dev	0.075	**0.066**	0.083	0.097	0.102	0.082	0.1	0.076	0.073	**0.066**
Annualized Sharpe	1.707	0.704	**2.499**	0.895	1.066	1.436	1.325	0.785	1.539	0.706
Skewness	-0.022	0.116	0.095	-0.98	-1478	-0.994	-1422	-0.466	-0.106	0.115
Excess Kurtosis	2.116	2.648	4.273	8.912	22.252	11.565	22.852	5.986	3.19	2.649
Maximum Drawdown	0.129	**0.104**	0.126	0.299	0.267	0.15	0.295	0.12	0.123	0.104
Historical VaR (95%)	-0.007	-0.007	-0.007	-0.009	-0.009	-0.007	-0.009	-0.007	-0.007	-0.007
Annualized Turnover	62.071	**0**	50.571	84.929	47.357	36.143	46.071	14.286	39.643	0.286
Maximum cost (%)	0.047	4.823	0.178	-0.009	0.025	0.06	0.069	-0.228	0.048	**7.112**

Table 2.3. *Out-of-sample investment performance of the machine learning algorithms with different risk aversion thresholds*

Overall, the annualized returns of the analyzed machine learning algorithms tend to increase with an increase in risk aversion threshold value with a peak at 0.9 and slightly decreasing back at 0.95. Likewise, with the exception of the C5.0 and GLM, the same behavior is observable at the level of the annualized Sharpe ratio. However, we note that both performance measures increase at the cost of an increased turnover. Furthermore, there is a strong link between the increase in threshold value and the decrease in

maximum drawdown, indicating a proper function of the safety valve. We could determine the optimal value of the threshold value based on its preferences for a real-life application of the presented framework.

The historical value at risk that represents the lower 5% quantile of historical returns appears to be stable over the different threshold values in the tactical asset allocation framework for the machine learning algorithms used. Their values also lie around 1% or lower, indicating a low risk of losing the invested wealth.

While analyzing the individual performance of the weighted function, it is clear that the gains, in terms of returns on investment as well as the Sharpe ratio, seem to be moderated and lie on the same scale as the medium performers. The weighted function appears to have a significant influence on tempering the maximum drawdown. This indicates the highest added value of the weighted version of the machine learning algorithm output.

It can be noted from Table 2.3 that the turnover is at its lowest with a risk aversion threshold of 0.5. The latter can be explained through the abundant amount of high probabilities outputted by the ML algorithms. In other words, since the overall level of output probabilities is relatively high, only few instances will be inferior to the threshold probability, thus causing few changes in the taken positions.

Recall that the maximum cost in percentage from Table 2.3 indicates the maximal transaction costs attached to a switch in the taken position, to safeguard the same annualized returns as the best-performing benchmark. Placing the risk aversion threshold value at the level of 0.5 results in the negative maximum cost for all analyzed ML algorithms. This configuration should thus not be used by investors. Note further that, with the exception of the strategies using C5.0 and the multilayer perceptron, we find positive values of the maximum cost that are lower than 0.1%. This implies that if investors are charged higher transaction costs than 10 basis points, the corresponding ML algorithm strategies would underperform the benchmark. However, the maximum cost attached to the multilayer perceptron ranges between 0.18% and 0.25%. As a result, investors using the multilayer perceptron strategy are more likely to outperform the benchmark in the presence of transactions cost.

By comparing the results of Table 2.3 with those obtained in Hull and Qiao (2015), we could quickly note the added value of using the ML algorithm in the context of tactical asset allocation. Hull and Qiao (2015) deploy a kitchen sink model that displays an annual return of 12.11%, a Sharpe ratio of 0.85 and a maximum drawdown of 21.12% over the period between August 6, 2001 and April 5, 2015. In terms of the individual performance of each algorithm, we observe in Table 2.3 that the C5.0 performs the worst. This can be explained by the statistical evidence that the most often predicted probability lies around 50%. When the risk aversion threshold increases, the ceiling of the strategy is reached by simply becoming the performance of the Treasury bond. The latter influences the displayed statistical performance of the algorithm, in which positions were solely taken on the riskless Treasury bond, indicating the flaws in the of market timing ability of the algorithm. Besides, increasing the risk aversion threshold probability positively impacts the performance of the linear discriminant analysis and the KNN, in terms of both return and Sharpe ratio. Moreover, this indicates the extreme nature of the outputted probability for the ML algorithms that can be caused by the strong class imbalance of the target variable. In addition, the KNN, generalized linear model and the Linear discriminant analysis appear to have the lowest varying performance caused by changes in the threshold probability.

Comparing the generalized linear model and elastic net regression, it appears that the penalty applied in the elastic net regression increases the predictive power and economic performance. The latter can be explained through the applied regularization that increases the bias of predictions in the training data, leading to a higher generalization power of the model on out-of-sample data. In other words, the regularization counters overfitting of a model on training data.

With the exception of a threshold probability of 0.5, both the random forest and elastic net regression indicate similar performances. Further, it is worth mentioning that the elastic net regression, random forest algorithm, the multilayer perceptron and the weighted function outperform the remaining machine learning algorithms. Overall, the MLP seems to be the best-performing model in terms of return on investment, Sharpe ratio, kappa statistic, accuracy and area under the curve.

In the remaining part of this section, we will take a closer look at the MLP. In Figure 2.3, we illustrate the behavior of the downward probability, predicted by the MLP for the out-of-sample period starting on December 2, 2005 until January 31, 2019. The black line represents the cumulative return on the SPY ETF, the red line illustrates the downward probability outputted from the ML algorithm and the blue line represents the downward probability from Faber's framework.

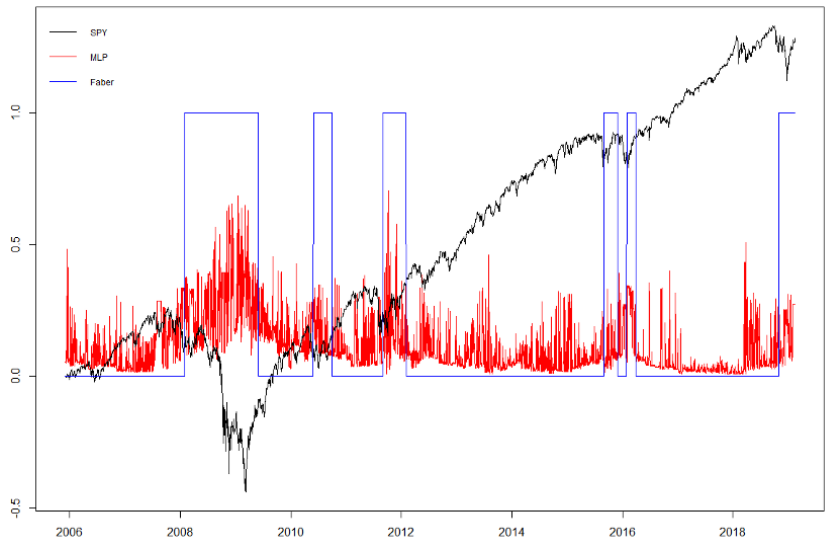


Figure 2.3. *Out-of-sample cumulative return of the SPY ETF (black line), downward probability predicted by the multilayer perceptron (red line) and Faber's timing to be invested in the market (blue line). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Evidently, we observe that the prediction made by the multilayer perceptron is negatively correlated with the behavior of the cumulative return of the SPY ETF. For instance, during the 2007–2009 financial crisis, we observe a significant increase in the extracted ML signal. Additionally, during the recent market selloffs in late 2018, we observe a significant relative increase in the ML signal. While the signal is relatively noisy, we discern that the relative levels are low during normal times versus times of market turmoil.

Interestingly, major downward movements of the SPY ETF are negatively correlated with the output generated by the implementation of the tactical asset allocation strategy, as proposed by Faber, with a delayed start. In other terms, the price of the SPY ETF undergoes a short period of decrease before the output of Faber's market timing strategy indicates a selling signal.

Furthermore, from the noisy output of the multilayer perceptron, as illustrated in Figure 2.3, we can easily see that the performance of tactical asset allocation strategy is highly dependent on the choice of the threshold value. The latter will be investigated from a closer perspective for the multilayer perceptron.

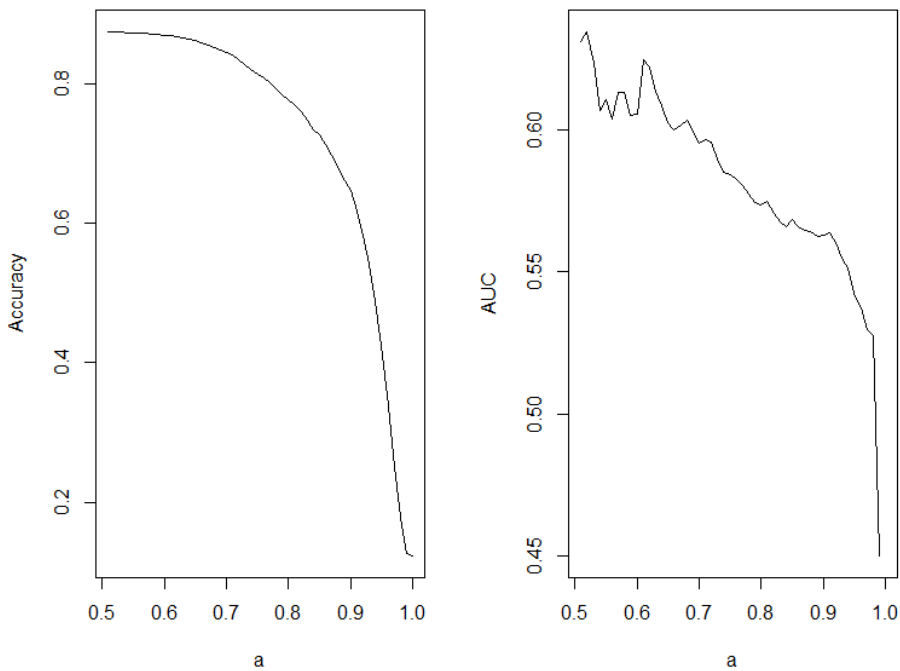


Figure 2.4. *Effect of variating risk aversion threshold value on accuracy and area under the curve (AUC) of output of the multilayer perceptron*

Figure 2.4 illustrates the behavior of the accuracy and area under the curve for variating threshold values a (from $a = 0.5$ to $a = 1$). It can be noted from Figure 2.3 that an increase in the threshold value results in a decrease

of the accuracy and area under the curve, for the output probabilities of the multilayer perceptron. Both curves thus exhibit a similar decreasing concave behavior with a smoother curve for the accuracy.

The reason behind this decreasing behavior lies in the scarcity of output probabilities reaching extreme values (close to 100%). In other words, increasing the risk aversion threshold causes an increase in instances that are classified as negative (or false negatives) which leads to a decrease in the accuracy and area under the curve. Note that we could expect from Figure 2.4 that the optimal risk aversion threshold value is 0.5.

From an investor's perspective, we would be interested in determining the financial implications of the strategy with respect to the predicting capabilities of the ML algorithms, as measured by the statistical criteria. Figure 2.5 illustrates the behavior of the Sharpe ratio and kappa statistic for variating risk aversion threshold.

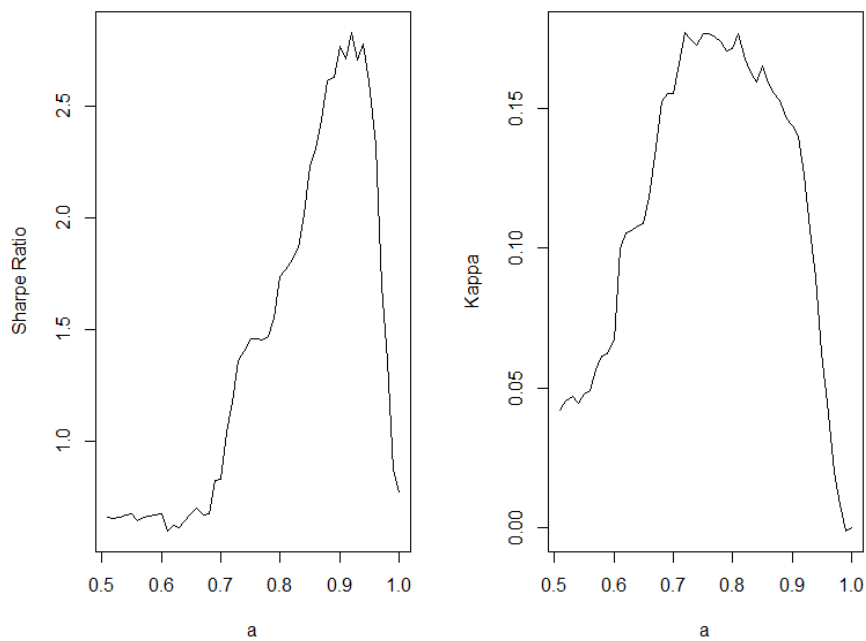


Figure 2.5. *Effect of variating the risk aversion threshold on the Sharpe ratio and kappa statistic of output of the multilayer perceptron*

Figure 2.5 illustrates the behavior of the Sharpe ratio and kappa statistic for variating risk aversion threshold a (from $a = 0.5$ to $a = 1$). In Figure 2.5, we can observe that the effect of changes in the threshold value exhibit similar behaviors with different maxima. Both the Sharpe ratio and kappa statistics appear to increase until reaching their respective maxima followed by a decrease.

From the comparison of Figures 2.4 and 2.5, it appears that the behavior of the risk-adjusted returns, with respect to variating threshold values, can best be approximated by the kappa statistic. Aiming for an ML algorithm that tries to maximize the accuracy or area under the curve does not seem to be optimal in terms of generated risk-adjusted returns for the multilayer perceptron, in the context of the tactical asset allocation strategy. In this optic, choosing a threshold value that maximizes the kappa statistic will result in a more profitable strategy, in terms of risk-adjusted returns compared to the other statistical criterions. Note that from the shape of the curves in Figure 2.5, it follows that we could use a predetermined value (e.g. 0.05) of the kappa statistics as the minimum requirement for determining the risk aversion threshold. Setting a minimum requirement on the kappa statistic would result in a strategy where the risk-adjusted returns are proportional to the choice of the threshold value.

By its inner construction, the tactical asset allocation framework allows the reduction of the overall error and cost attached to misclassification down to false positives, causing the negative returns of the strategy. All false negatives are in this optic hedged in the tactical asset allocation framework by investing in the riskless Treasury bond ETF, guaranteeing a tempered return. In other words, it is crucial for an ML algorithm used in the tactical asset allocation to lower the amount of false positives to generate a usable strategy. Note that this effect is the major reason why discrepancies are generated between the behavior of the Sharpe ratio as presented in Figure 2.5 and the accuracy displayed in Figure 2.4.

The ML-based investment algorithms have a computational cost with respect to more simple benchmark strategies, like buy-and-hold or constant mix strategies. Table 2.4 compares the performance of the benchmarks with the performance of the multilayer perceptron with risk aversion thresholds of

0.9 and 0.95. These benchmarks constitute the buy-and-hold investments in the SPY ETF and the 7–10 year Treasury bond ETF, Faber's tactical asset allocation framework and the naive benchmark of 60/40 investing, as described in section 2.2.4.

	SPY	IEF	60/40	Faber	MPL (a=0.9)	MPL (a=0.95)
Annualized Return	0.082	0.046	0.074	0.11	0.29	0.206
Annualized Std Dev	0.191	0.066	0.106	0.129	0.103	0.083
Annualized Sharpe	0.431	0.704	0.7	0.852	2.822	2.499
Skewness	0.169	0.116	0.359	0.53	0.99	0.095
Excess Kurtosis	15.755	2.648	17.425	5.127	15.541	4.273
Maximum Drawdown	0.552	0.104	0.314	0.173	0.274	0.126
Historical VaR (95%)	0.018	0.007	0.01	0.013	0.009	0.007

Table 2.4. *Investment performance of the analyzed benchmarks*

Overall, Table 2.4 indicates that Faber's tactical asset allocation strategy outperforms the other benchmarks in terms of annualized return and Sharpe ratio. With the exception of the Treasury bonds, Faber's framework has the smallest maximum drawdown of all benchmarks, implying a lower downside risk relative to the SPY and the naive 60-40 strategy. The outperformance of Faber's framework is further illustrated in Figure 2.6, which demonstrates the cumulative return of all benchmarks. Consistent with Figure 2.6, we observe that Faber's strategy switches to Treasury bonds during the 2007–09 financial crisis, resulting in the best-performing benchmark in terms of cumulative return over the sample period.

To put the results of the benchmarks into perspective, the presented results of the MLP algorithm with risk aversion thresholds of 0.9 and 0.95, we can clearly outperform all benchmarks in terms of annualized – and risk-adjusted returns. In its configuration with a risk aversion threshold of 0.95, even Faber's framework is outperformed in terms of maximum drawdown, indicating the effectiveness of the safety valve function of the risk aversion threshold.

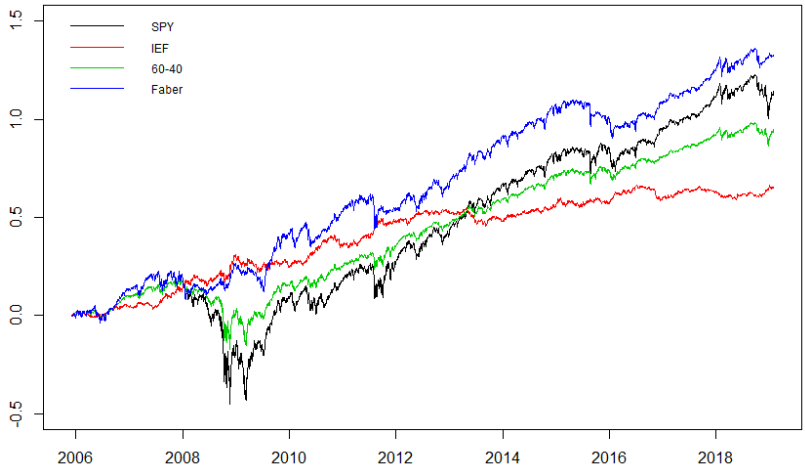


Figure 2.6. *Cumulative return of the benchmarks – SPY and IEF, respectively, refer to the return on the S&P 500 and the 7–10 year Treasury bond ETFs. The 60-40 green line denotes naive benchmark. Faber refers to the strategy proposed by Faber (2007) that rotates between the SPY and IEF based on the 200-day moving average. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

2.5.4. *The best performer, benchmark and model average*

In line with Figures 2.4 and 2.5, Figure 2.7 illustrates the cumulative return of the strategy, based on the output signal of the MLP algorithm with risk aversion thresholds of 0.95 and 0.90 (black line). The ML strategy is compared with Faber's benchmark over the same period (red line). In addition, we add the two weighted ML outputs, one with equal weights (green) and the other (blue) with respect to the sum squarer error as described in equation [2.9].

In either case, we observe that the MLP strategy outperforms the benchmark in terms of cumulative return. At the same time, we observe that Faber's strategy outperforms the average ML strategies. The latter evidence undermines the usefulness of ML, compared to an ad hoc technique that relies on basic computations. Additionally, the higher risk-taking profile of the $a = 0.9$ configuration leads to a higher maximum drawdown of 27.3% compared to the $a = 0.95$ which is 12.6%. Overall, the MLP strategy configured with a threshold value of 0.9 is more sensitive to the 2007–2009 financial crisis than the $a = 0.95$ counterpart. While the strategy has done

well since the start of the crisis, we observe that it does underperform towards the end of 2009.

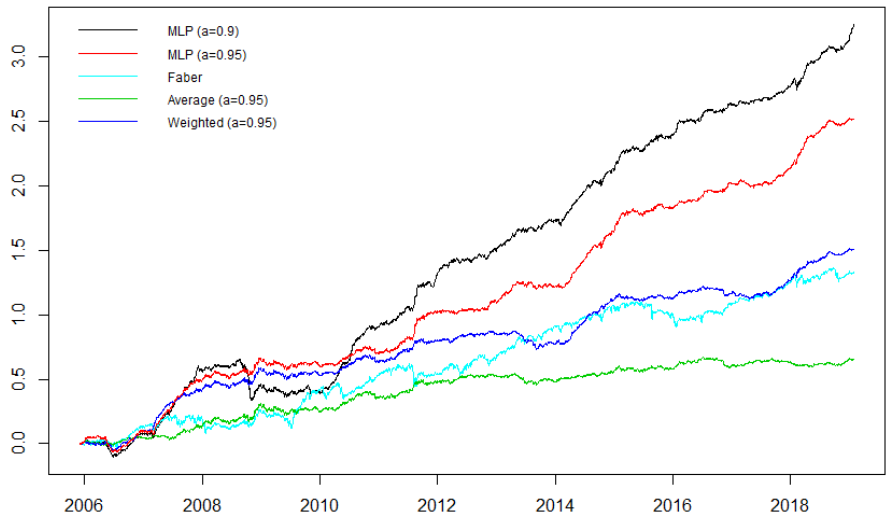


Figure 2.7. *Out-of-sample return of the strategy from the multilayer perceptron, Faber's framework and the model averaging strategy with risk aversion threshold probability of 0.95. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Note that after the period of instability, the strategy then shows a tremendous increase in wealth leading to an accumulated annualized return of 28.9% over the analyzed period. An important caveat is of course that the performance achieved is conditional on the chosen threshold return value τ_t, probability classification threshold value a, the size of the training and evaluation sample and the pseudo-random seeds taken.

2.5.5. *LIME*

We now use LIME (Ribeiro, 2016) to uncover the more important traits leading to the MLP performance. As explained in section 2.4.3, it uses a local linear approximation to derive the important traits affecting the decision. The results are shown in Figure 2.8, where we show the averaged relative effect on the MLP decision with $a = 0.9$.

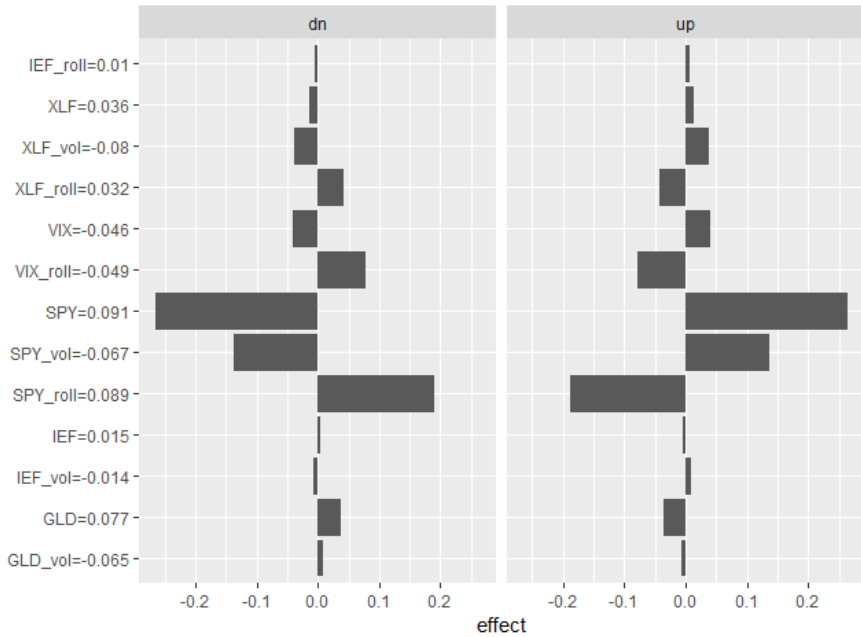


Figure 2.8. *Average effect of averaged input features*

It is evident from Figure 2.8 that the SPY-related features are the most prominent influencers of predicted probability. The SPY related features are composed of SPY return, the MA of SPY return and the SPY volume. Both the SPY return and volume exert a similar effect on the predictions made by the multilayer perceptron. While usually it is unclear whether past prices and high volume imply a future market increase or drop, the ML indicates that it has a positive effect. In other words, Figure 2.8 shows that an increase in the SPY return and volume is associated with a future price increase. The opposite logic holds true for the MA of the SPY return. This indicates that the interaction between a longer-term price behavior of the SPY ETF and a short-term price behavior of the SPY ETF exhibits opposite effects on the predicted probabilities, made by the multilayer perceptron. The presented interactions also imply that a low (high) return on the SPY ETF along with a high (low) return on a rolling basis indicates a higher probability of downward (upward) movement. The latter suggests a trend-following behavior, such that the longer-term past level of the SPY ETF (the SPY on a rolling basis) and the short-term past level of the SPY ETF, determine the direction of the future trend.

A caveat is that using this approach under the aforementioned conditions only hints at how predictions are made on average. To have a more accurate linear approximation the user should perform the same analysis over all different local predictions individually.

2.6. Conclusion

This chapter illustrates the use of open data and software for the investors who are interested in tactical asset allocation using machine learning, without spending budget on data or software acquisition. We show the potential gains in performance achieved through machine learning as compared to the long-only, constant mix and price momentum strategies.

Our analysis is suggestive and by no means exhaustive. The quest to find the optimal machine learning algorithm with the adequate hyperparameters is a difficult task to perform, partially due to the noise included in financial time series. Improved data input, data processing, calibration of the hyperparameters and the use of other machine learning algorithms are all promising avenues to further improve the performance.

2.7. Acknowledgments

The authors thank David Ardia, Keven Bluteau, Andres Algaba, Emmanuel Jurczenko (the editor) and Sam Verboven for their constructive suggestions. The authors also thank the conference participants at the DSF-R conference in Vienna, the 2019 R in Finance conference in Chicago and the 2019 High Frequency Finance conference at Stevens Institute of Technology, for their valuable feedback.

2.8. References

Bacon C.R. (2008). *Practical at Portfolio Performance Measurement and Attribution*. John Wiley & Sons, Chichester.

Ben-David A. (2008). About the relationship between ROC curves and Cohen's kappa. *Engineering Applications of Artificial Intelligence*, 21(6), 874–882.

Bergmeir C.N. and Benítez Sánchez J.M. (2012). Neural networks in R using the Stuttgart neural network simulator: RSNNS. American Statistical Association.

Boudt K., Darras J., Nguyen G.H. and Peeters B. (2015). Smart beta equity investing through calm and storm. In *Risk-Based and Factor Investing*, Jurczenko E. (ed.). ISTE Ltd, London and Elsevier, Oxford.

Branco P., Torgo L. and Ribeiro R.P. (2016). A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2), 31.

Breiman L. (2001). Random forests. *Machine learning*, 45(1), 5–32.

Campbell J.Y. and Shiller R.J. (1988). The dividend-price ratio and expectations of future dividends and discount factors. *The Review of Financial Studies*, 1(3), 195–228.

Cochrane J.H. (2007). The dog that did not bark: A defense of return predictability. *The Review of Financial Studies*, 21(4), 1533–1575.

Cohen J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1), 37–46.

Cortes C. and Vapnik V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

DeMiguel V., Garlappi L., and Uppal R. (2007). Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy? *The Review of Financial Studies*, 22(5), 1915–1953.

DeMiguel V., Plyakha Y., Uppal R. and Vilkov G. (2013). Improving portfolio selection using option-implied volatility and skewness. *Journal of Financial and Quantitative Analysis*, 48(6), 1813–1845.

Faber M. (2007). A quantitative approach to tactical asset allocation. *The Journal of Wealth Management*, 9(4), 69–76.

Fama E.F. and French K.R. (1988). Dividend yields and expected stock returns. *Journal of Financial Economics*, 22(1), 3–25.

Fisher R.A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2), 179–188.

Friedman J., Hastie T. and Tibshirani R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1), 1–22.

Fukunaga K. (2013). *Introduction to statistical pattern recognition*. Elsevier, Oxford.

Hull B. and Qiao X. (2017). A practitioner's defense of return predictability. *The Journal of Portfolio Management*, 43(3), 60–76.

Jiang F., Lee J., Martin X. and Zhou G. (2019). Manager sentiment and stock returns. *Journal of Financial Economics*, 132(1), 126–149.

Karatzoglou A., Smola A., Hornik K. and Zeileis A. (2004). kernlab-An S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9), 1–20.

Kritzman M., Page S. and Turkington D. (2012). Regime shifts: Implications for dynamic strategies (corrected). *Financial Analysts Journal*, 68(3), 22–39.

Kuhn M. (2008). Building predictive models in R using the caret package. *Journal of Statistical Software,* 28(5), 1–26.

Kuhn M. and Johnson K. (2013). *Applied Predictive Modeling*. Springer, New York.

Kohavi R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. *Ijcai*, 14(2), 1137–1145.

Liaw A. and Wiener M. (2002). Classification and regression by randomForest. *R news*, 2(3), 18–22.

Mandrekar J.N. (2011). Measures of interrater agreement. *Journal of Thoracic Oncology*, 6(1), 6–7.

Meucci A. (2012). Effective number of scenarios in fully flexible probabilities. *GARP risk professional*, 34–37.

Mohamad I.B. and Usman D. (2013). Standardization and its effects on K-means clustering algorithm. *Research Journal of Applied Sciences, Engineering and Technology*, 6(17), 3299–3303.

Molnar C. (2018). *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. Leanpub, Canada.

Molnar C., Bischl B. and Casalicchio G. (2018). iml: An R package for Interpretable Machine Learning. Journal of Open Source Software, 3(26), 786.

Nelder J.A. and Wedderburn R.W. (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3), 370–384.

Peterson B.G. and Carl P. (2018). PerformanceAnalytics: Econometric tools for performance and risk analysis. *R package version* 1.5.2.

Quinlan J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann. Los Altos, California.

R Core Team. (2018). R: A Language and Environment for Statistical Computing. *R Foundation for Statistical Computing*. Vienna, Austria.

Ribeiro M.T., Singh S. and Guestrin C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 1135–1144.

Rosenblatt F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386.

Rumelhart D.E., Hinton, G.E. and Williams R.J. (1985). *Learning Internal Representations by Error Propagation*, No. ICS-8506, 1, 318–362. (Institute for Cognitive Science, University of California, San Diego).

Ryan J.A. and Ulrich J.M. (2008). quantmod: Quantitative financial modelling framework. R package version 0.3-5. Available at: http://www.quantmod.com and http://r-forge. r-project. org/projects/quantmod.

Segal J. (2019). AQR's problem with machine learning: CATS morp into dogs. Available at: https://www.institutionalinvestor.com/article/b1fsn64kfq8b5h/AQR-s-Problem-With-Machine-Learning-Cats-Morph-Into-Dogs.

Shannon C.E. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(3), 379–423.

Stock J.H. and Watson M.W. (2004). Combination forecasts of output growth in a seven country data set. *Journal of Forecasting*, 23(6), 405–430.

Venables W.N. and Ripley B.D. (2002). *Modern Applied Statistics with S*. Fourth Edition. Springer, New York.

3

Sparse Predictive Regressions: Statistical Performance and Economic Significance

We propose and evaluate a variety of penalized regression methods for forecasting and economic decision-making in a data-rich environment under parameter uncertainty. Empirically, we explore the statistical and economic performance across different asset classes such as stocks, bonds and currencies, and alternative strategies within an asset class (e.g. momentum and value in the space of equity). The main result shows that penalties that both shrink the model space and regularize the remaining regression parameters, for example, elastic net penalty, tend to outperform competing sparse and dense methodologies, both statistically and economically.

3.1. Introduction

The recent advancements in the fields of econometrics, statistics, and machine learning, combined with the increasing availability of large datasets, have spurred the interest of financial economists in predictive models that can handle many explanatory variables. However, not all predictors are necessarily relevant. As a result, decision-makers often pre-select the most important candidate predictors by appealing to the existing academic literature, as well as to established economic theories. Nevertheless, decision-makers are left with tens, if not hundreds, of possibly relevant predictors, leaving the econometrician, the investor or the policy maker with the daunting task of predicting asset returns in a data-rich environment.

Chapter written by Daniele BIANCHI and Andrea TAMONI.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Importantly, the out-of-sample performance of standard techniques, such as ordinary least squares, maximum likelihood, or Bayesian inference with uninformative priors, tends to deteriorate as the number of predictors increases (relative to the number of observations), a fact known as the curse of dimensionality.

A class of models that has been proved effective in dealing with the curse of dimensionality within a linear setting is the penalized regression model. In its general form, a penalized regression is a linear model that is penalized for having too many variables in the model by adding a constraint, i.e., a penalty term, to the loss function. Depending on the functional form of the penalty term, the penalized regression either shrinks the variable space – by setting some of the regression coefficients to zero – or regularizes the beta estimates – by smoothly shifting the estimates towards small values close to – but not exactly equal to – zero. The most common example of a shrinkage linear regression is the Lasso, originally proposed by Tibshirani (1996). A common example of a regularized regression model is the so-called Ridge regression, originally introduced by Hoerl and Kennard (1970). Unlike in the case of lasso, which employs an L1-norm penalty, the ridge penalty is based on an L2-norm which precludes shrinkage to zero: in ridge regressions, variables with minor contribution to the outcome have a loading close to zero, despite none of them being set exactly to zero. That means that while the lasso is a sparse regression model, the ridge still falls within the sub-class of dense linear regression models (see Giannone *et al*. 2018 for a discussion).

In this chapter, we will evaluate the ability of a large set of penalized regression models to predict excess returns within a data-rich environment. Predicting asset returns plays a central role in empirical finance. Recent work by Kozak *et al*. (2019) and Gu *et al*. (2018) shows the promise of machine learning methods, such as traditional regularization techniques and trees, to deal with this challenge. To date, however, the empirical evidence on new methodologies to forecast returns is mostly focused on individual US equities[1]. Thus, this chapter investigates the potential benefits of using penalized regression methods to forecast returns in different asset classes

1 Notable exceptions are Bianchi *et al*. (2019a) who investigate the role of a variety of machine learning methods to forecast Treasury bond returns, Feng *et al*. (2018) and Rossi (2018), who applies Deep Learning and Boosted Regression Trees to forecast the S&P 500 market returns.

such as equities, currencies and Treasury, corporate, and sovereign bonds, and for different strategies within an asset class, for example, momentum and value portfolio returns in equities and currencies.

We restrict our analysis to a *linear* framework on purpose. By doing so, our results (1) are more directly comparable to the standard approach based on a linear predictive system in a small data setting (see, for example, Barberis, 2000; Pastor and Stambaugh, 2009); (2) isolate the advantages of supervised machine learning methods relative to classical (linear and unsupervised) approaches like principal component regressions that have been commonly used to deal with data-rich environments.

To predict returns, we make use of both asset-specific predictors and a large set of economic variables from the FRED-MD database. The FRED-MD database includes numerous macro variables in diverse categories, namely: output, income, and the labor market; consumption, orders, and inventories; money and credit; yields and exchange rates; housing and inflation. Hence, this large set of variables captures several potentially relevant risks in the macroeconomy. In addition to these aggregate macroeconomic variables, we also consider a set of aggregate financial variables, for example dividend–payout ratio, default spread and net-equity issuance, taken from Welch and Goyal (2007). The asset-specific predictors, on the other hand, are justified by accounting identities based on log-linearization of returns à la Campbell and Shiller (1988). The spirit of the exercise is to mimic an investor/decision-maker who has both aggregate macro and financial information, as well as asset-specific predictors, to construct its expected future excess returns.

Methodologically, we adopt a Bayesian estimation framework for each penalized regression specification (see, for example, Park and Casella, 2008; Hans, 2009; Carvalho *et al*., 2010; Kyung *et al*., 2010, among others). The advantages of Bayesian methods with respect to standard methodologies are several: first, penalty terms can be naturally expressed as shrinkage priors. By doing so, the penalized regression model can be cast into an encompassing hierarchical model which allows us to estimate, in a single step, both the penalty and the regression parameters. This fact is especially useful when both global and local, i.e., predictor-specific, penalty parameters need to be estimated, since sequential cross-validation procedures tend to induce too much shrinkage (see, for example, Zou and Hastie 2005). Second, the

Bayesian framework explicitly accounts for parameter uncertainty, and naturally generates confidence intervals for the parameters of interest, something that is not easily obtained in a frequentist setting. In fact, Kyung *et al*. (2010) show that classical penalized regressions procedures can result in poorly performing variance estimates. Third, Bayesian estimates rely on Markov chain Monte Carlo (MCMC) sampling, which, unlike standard optimization procedures, is more robust to noisy signals, observation outliers, as well as multiple modes that are commonly encountered with non-convex penalties.

We explore both the econometric underpinnings of each methodology and their economic gains across different asset classes. The main results show that penalty terms that *shrink* the model space – by setting some regression coefficients to zero – *and regularize* the remaining regression coefficients tend to outperform both dense (e.g. ridge regression) and sparse (e.g. lasso regression) methodologies. Such penalized regression is known as elastic net and has been introduced by Zou and Hastie (2005). In particular, we show, via a simple rotation trading strategy, that an investor would be willing to pay a positive fee to have access to the predictive content of the elastic net on a given portfolio, as opposed to relying on a naive historical mean estimate as the main trading signal. Finally, we show that the amount of shrinkage (or, equivalently, the amount of sparsity) tends to evolve over time and differs across asset classes. That means that assuming sparsity to be constant in the data generating process may be costly in terms of out-of-sample performance, both statistically and economically (see Bianchi *et al*. 2019b for an in-depth discussion of this point).

The rest of this chapter is organized as follows. Section 3.2 discusses the use of machine learning in financial economics. Section 3.3 describes the data with particular emphasis on the various asset classes and the strategies investigated in our empirical study. Section 3.4 provides an overview of the shrinkage priors adopted in our analysis. Section 3.5 reports our results on predicting returns in various asset classes.

3.2. Related literature

This chapter contributes to a growing literature that explores the use of machine learning methodologies for returns predictability. The literature on

time-series return predictability is too large to be reviewed here. Thus, in this chapter, we try only to position our contribution within the continuously growing recent literature on machine learning and asset pricing.

Early attempts to explore machine learning in empirical finance are Kuan and White (1994), Lo (1994), Hutchinson *et al*. (1994) and Yao *et al*. (2000), who introduced the use of artificial neural networks. More recent works have applied machine learning techniques in cross-sectional return prediction. These papers focus on predicting monthly US equity returns based on lag firm characteristics and, to a lesser extent, macro predictors[2]. Differently from these papers, we forecast anomaly equity portfolios returns such as value and momentum. In this respect, our analysis complements the literature on factor timing that uses valuation ratios to forecast portfolio returns (see, for example, Asness *et al*., 2000; Cohen *et al*., 2003; Baba-Yara *et al*., 2018; and Arnott *et al*., 2019). Our work is also related to Nalbantov *et al*. (2006) who apply support vector regressions with technical and economic variables to time the size and value premium in the USA. Two recent papers forecast Treasury bond returns using group-lasso (Huang and Shi 2018), and a novel type of neural network dubbed "group ensemble" (Bianchi *et al*. 2019a), and show that there is information in macro variables, which is not contained in yields that is useful to predict bonds. These papers forecast individual maturity bond returns, whereas we instead forecast an average of excess returns across maturities that has been duration standardized to avoid overweighting particular maturities.

Our investigation of predictability for the aggregate US equity premium is mostly inspired by the work of Welch and Goyal (2007) and Rapach *et al*. (2013)[3]. Welch and Goyal (2007) using show that a large set of economic variables leads to out-of-sample performance which is hard to outperform using historical means. However, Rapach *et al*. (2013) show that combining individual model forecasts improves the out-of-sample equity premium

2 See, for example, Feng, Giglio and Xiu (2019), Freyberger *et al*. (2017), Giannone *et al*. (2018), Giglio and Xiu (2017), Heaton *et al*. (2017), Kozak *et al*. (2019), Feng, Polson and Xu (2019), Gu *et al*. (2018), Kelly *et al*. (2018), Messmer (2018) and Chen *et al*. (2019).

3 See Koijen and Van Nieuwerburgh (2011) and Rapach and Zhou (2013) for an overview of the literature on stock return forecasting, highlighting the challenges faced by forecasters as well as strategies for improving return forecasts. See also Binsbergen *et al*. (2010) for the importance of present-value constraints in the context of predicting the aggregate US equity market.

prediction. Rapach *et al*. (2013) use elastic net to study the role of lagged US returns in forecasting non-US industrialized countries' equity returns. Furthermore, Feng *et al*. (2018) and Rossi (2018) have recently emphasized the role of non-linearities for predicting aggregate stock returns. None of these papers has studied the role of the FRED-MD set of macroeconomic variables for predicting returns. Moreover, our analysis of robust shrinkage priors, like the horseshoe, and group-based methodologies, like the group-lasso, for the predictability of asset returns, is new to the literature.

Finally, our analysis of predictability of value and momentum strategies for currencies complements the analysis of Baba-Yara *et al*. (2018), who show that the value spread forecasts value returns in currency markets[4].

3.3. Data: portfolios and predictors

In this section we provide details on the asset returns and the predictors used in our empirical investigation. Throughout, we use monthly data and we forecast one-month holding-period returns. The sole exception is Treasury bonds, for which we focus on one-year holding-period returns. Next, we describe the return strategies used in each asset class:

– Aggregate equity market: we use S&P 500 index returns from 1963 to 2016 from Center for Research in Security Press (CRSP) month-end values. Stock returns are the continuously compounded returns on the S&P 500 index, including dividends. Excess returns are above the US Treasury bill rate from Ken French's data library.

– Equity portfolios: we forecast value and momentum returns. To construct value returns, we form value-weighted decile portfolios each month based on (the cross-sectional distribution of) the value measure (see below). The US individual stock data is from CRSP and Compustat. Following Asness *et al*. (2013), we only include in our value and momentum strategies those stocks that cumulatively account for 90% of the total market capitalization in CRSP. We limit the analysis to a sample from January 1963 to December 2016.

– Currencies: we forecast value and momentum returns. To construct value returns, we form an equal-weighted high and low portfolio by splitting

4 Lustig *et al*. (2014) shows that the dollar's risk premium varies strongly and counter-cyclically.

the currencies at the median of ranked values (based on the cross-sectional distribution of the value measure, see below). We obtain spot and forward exchange rates for Australia, Canada, Germany (spliced with the Euro), Japan, New Zealand, Norway, Sweden, Switzerland, the UK and the United States. The sample period for currencies runs from February 1976 to December 2016.

– Treasury bonds: we obtain yields on zero-coupon Treasury bonds with maturities from 1 to 15 years from Gurkaynak *et al*. (2007). We calculate log excess returns from these log yields. As standard for one-year holding period returns, we use the one-year zero-coupon yield as the risk-free rate. We then rescale the excess returns of bonds of maturity n, $rx_{t+1}^{(n)}$, by dividing them by $n-1$:

$$rx_{t+1}^{(n)} = -y_{t+1}^{(n-1)} + \frac{n}{n-1} y_t^{(n)} - \frac{1}{n-1} y_t^{(1)} \ .$$

The transformed excess returns all have modified duration equal to unity. This leads our scaled returns to have approximately equal standard deviation, whereas this statistic varies by a factor of 10 for unscaled returns.

– Corporate bonds: corporate bond prices come from the Lehman Brothers Fixed Income Database, the Mergent FISD/NAIC Database, TRACE, and DataStream, which together provide an extensive data set on publicly traded corporate bonds from 1973 to 2016. Let $P_{i,t}$ be the price per one dollar face value for corporate bond i at time t including accrued interest, and $C_{i,t}$ be the coupon payment. The return on the bond is then: $R_{i,t+1} = \frac{P_{i,t+1}+C_{i,t+1}}{P_{i,t}}$. Suppose that there is a matching Treasury bond for corporate bond i, such that the matching Treasury bond has an identical coupon rate and repayment schedule to corporate bond i. Let $P_{i,t}^f$ and $C_{i,t}^f$ be the price (including accrued interest) and coupon rate, respectively, for such a Treasury bond. The return on the matching Treasury bond is then: $R_{i,t+1}^f = \frac{P_{i,t+1}^f+C_{i,t+1}^f}{P_{i,t}^f}$.

We follow Nozawa (2017) and define the log return on corporate bond i, in excess of the log return on the matching Treasury bond,

$$r_{i,t+1}^e \equiv \log R_{i,t+1} - \log R_{i,t+1}^f.$$

Finally, we compute the equal-weighted market portfolio returns (in excess of log Treasury returns) according to

$$r_t^e = \frac{1}{N_t} \sum_{i=1}^{N_t} r_{i,t}^e.$$

– Sovereign: We follow Borri and Verdelhan (2010) and focus on emerging countries that are included in JP Morgan's emerging market bond index (EMBI)[5]. Let r_i^e denote the log excess return of a US investor who borrows funds in US dollars at the risk-free rate r^f in order to buy country i's EMBI bond, then sells this bond after one month, and pays back the debt. The log excess return is equal to: $r_{i,t+1}^e = p_{i,t+1} - p_{i,t} - r_t^f$, where $p_{i,t}$ denotes the log market price of an EMBI bond in country i at date t. Next, define the bond beta ($\beta_{i,EMBI,t}$) of each country i as the slope coefficient in a regression of EMBI bond excess returns on US equity excess returns: $r_{i,t}^e = \alpha_i + \beta_{i,EMBI,t} r_t^e + \varepsilon_t$, where r_t^e denotes the log total excess return on the MSCI US equity index. Borri and Verdelhan (2010) build portfolios of sovereign bond indices by sorting all countries in the sample in two groups on the basis of their bond betas. Then, they sort all countries within each of the two groups into three portfolios ranked from low to high probabilities of default; portfolios 1, 2 and 3 contain countries with the lowest betas, while portfolios 4, 5 and 6 contain countries with the highest. In our analysis, we focus on portfolios 1 and 4, that contain countries with the lowest default probabilities. The sample is from January 1995 to December 2016.

We use a large set of 120 macro variables from the FRED-MD database (McCracken and Ng 2016) to forecast returns[6]. This set is common across asset classes and strategies. The variables cover a wide array of categories (output and income; labor market; housing; consumption, orders and inventories; money and credit; interest and exchange rates; prices); as such, they capture

5 The JP Morgan EMBI Global total return index is a market capitalization-weighted aggregate of US dollar-denominated Brady Bonds, Eurobonds, traded loans, and local market debt instruments issued by sovereign and quasi-sovereign entities and mostly uncollateralized. See Borri and Verdelhan (2010) for additional information.

6 FRED-MD is available at Michael McCracken's webpage at https://research.stlouisfed.org/econ/mccracken/fred-databases/.

much of the macro information available to investors. We augment the FRED-MD dataset with asset-class specific predictors, which we describe next.

– Aggregate equity market: We follow Welch and Goyal (2007) and use as independent variables the dividend payout ratio (log dividend-earnings, d/e), the book-to-market ratio (b/m), the net equity expansion (ntis), the default return spread (def), and the stock variance (svar). The data are from Amit Goyal's website[7].

– Equity portfolios: We follow Asness *et al*. (2000), Cohen *et al*. (2003), and Baba-Yara *et al*. (2018), and use a measure of value to forecast value and momentum portfolios. To measure value for each firm i, we use the ratio of the book value to the market value of equity, or book-to-market ratio $BM_{i,t}$, as in Fama and French (1992). Book values are observed each June and refer to the previous fiscal year end. Market values are updated monthly, as in Asness and Frazzini (2013). Consistent with previous literature, we exclude financial firms: a given book-to-market ratio might indicate distress for a non-financial firm. Note that the use of value to forecast equity portfolios can be easily motivated using the log-linear present value model employed in Vuolteenaho (2002).

– Currency portfolios: We follow Baba-Yara *et al*. (2018) and use a measure of value to forecast value and momentum portfolios. In line with Asness *et al*. (2013), we measure value using long-term past returns. Specifically, we use the five-year change in relative purchasing power parity, which is calculated as the negative of the five-year spot return adjusted by the five-year foreign−US inflation difference. Currency value is large when the foreign currency has weakened relative to the dollar. As noted in Menkhoff *et al*. (2016), using five-year changes avoids potential problems arising from nonstationarity and base-year effects. Finally, note that the standard present-value formulation of Froot and Ramadorai (2005) shows that expected currency returns are a key driver of the real exchange rate. This motivates using real exchange rates as a measure of value for currencies.

– Treasury bonds: we define log forward rates as $f_t^{(n)} = p_t^{(n-1)} - p_t^{(n)}$, where $p_t^{(n)}$ denotes the time t log price of an n-year bond. We use only the

7 Compared to Welch and Goyal (2007) a few variables were excluded from the analysis. For example, we excluded the CAY variable, since this is the only available quarterly frequency. Variables like inflation and T-bill rate are already included in the FRED-MD database.

forward rates with maturity one to five years as predictive variables.

– Corporate bonds: Nozawa (2017) shows that the following present value identity holds

$$s_{i,t} \approx \sum_{j=1}^{T_i-t} \rho^{j-1} r^e_{i,t+j} + \sum_{j=1}^{T_i-t} \rho^{j-1} l_{i,t+j} + \text{const},$$

where T_i is the maturity of bond i, and

$$s_{i,t} \equiv \begin{cases} \log \frac{P^f_{i,t}}{P_{i,t}} & \text{for } t < t_D \\ 0 & \text{otherwise} \end{cases}$$

$$l_{i,t} \equiv \begin{cases} \log \frac{P^f_{i,t}}{P_{i,t}} & \text{for } t = t_D \\ 0 & \text{otherwise} \end{cases}$$

and t_D is the time of default. The variable $s_{i,t}$ measures credit spreads, while $l_{i,t}$ measures the credit loss upon default. To forecast the equal-weighted market portfolio returns, we use the equal-weighted average of the "price spreads", denoted by $s_t = \frac{1}{N_t} \sum_{i=1}^{N_t} s_{i,t}$.

3.4. Econometric framework

A decision-maker is interested in predicting some quantity of interest y based on a p-dimensional vector of predictors $\boldsymbol{x}$ which are all considered relevant *a priori*, but with some uncertainty on their actual predictive power. The conventional approach would be a generic time-series univariate regression of the form

$$y_t = \mu + \boldsymbol{\beta}' \boldsymbol{x}_{t-1} + \epsilon_t, \qquad [3.1]$$

where $\boldsymbol{\beta}$ is a p-dimensional vector of regression coefficients and $\epsilon_t \sim \pi\left(0, \sigma^2\right)$ with σ^2 a scalar regression variance. The vector $\boldsymbol{x}_t$ could contain dummies and, exogenous predictors, as well as lags of the target variable y_t. Despite its simplicity, such a linear predictive regression framework is arguably still the benchmark for financial economists and practitioners who often need to communicate intuitively the predictive content of a given variable to clients, investors and policy makers.

In many important financial applications, the dimension of the vector $\boldsymbol{x}_t$ used to make an informed decision is large, possibly so large to forbid the use of ordinary least squares (OLS) and/or maximum likelihood methods. In fact, at least *a priori*, all of these predictors could provide relevant information. In many such cases, we could easily get into a situation where the ratio p/T rapidly approaches 1, which, in turn, causes standard estimators such as OLS to overfit. That is, when faced with a large set of predictors and a small number of degrees of freedom, unbiased estimators (e.g. OLS) tend to have a very large variance. In such cases, introducing a bias on purpose through a shrinkage and/or regularization of the model space can help achieve a lower out-of-sample mean predictive squared error (MSPE).

For this reason, penalized regression methods such as the lasso (Tibshirani 1996), ridge regression (Hoerl and Kennard 1970), and elastic net (Zou and Hastie 2005) have been introduced over the years. The central idea of penalized regressions is to add a penalty term to the minimization of the loss function, with the explicit goal of shrinking small coefficients towards zero while leaving large coefficients intact, i.e.[8]

$$\mathcal{L}(\boldsymbol{\beta};\cdot) = \underbrace{\mathcal{L}(\boldsymbol{\beta})}_{\text{Loss Function}} + \underbrace{\phi(\boldsymbol{\beta};\cdot)}_{\text{Penalty Term}} . \qquad [3.2]$$

The loss function is commonly represented by the sum of squared residuals, i.e., $\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{2T}\|\boldsymbol{y} - \mu\mathbf{1} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2$, where $\boldsymbol{y} = (y_1, \cdots, y_T)'$ and $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T)'$. Depending on the functional form of the penalty term, the regression coefficients can be regularized and shrunk towards zero, completely set to zero, or a combination of the two. The penalized minimization problem outlined in equation [3.2] can be cast in a Bayesian framework by using a specific prior, depending on the type of penalization required. Formally, consider the following sampling distribution of the data given the predictors

$$\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\beta}, \mu, \sigma^2 \sim N\left(\mathbf{1}\mu + \boldsymbol{X}\boldsymbol{\beta}, \sigma^2\mathbb{I}_p\right), \qquad [3.3]$$

8 Note that the intercept is not included in the penalty terms. Penalization on the intercept would make the optimization procedure dependent on the initial values chosen for the asset returns, i.e. adding a fixed constant to the asset excess returns would not simply result in a shift of the prediction by the same amount.

where $\mathbf{1}$ is a vector of ones of size T, $\mathbb{I}_p$ is a p-dimensional identity matrix and $\sigma^2 \sim \pi\left(\sigma^2\right) d\sigma^2$. The Bayesian penalized regression can be expressed as a hierarchical model in which the sampling distribution [3.3] is coupled with a set of particular prior distributions for the regression coefficients. More specifically, the statistical representation of the priors is constructed from a scale mixture of normals, i.e. a Gaussian distribution is "mixed" or averaged over a general mixing distribution (see, for example, Andrews and Mallows 1974),

$$\beta_j|\lambda_j^2, \tau^2, \sigma^2 \sim N\left(0, \lambda_j^2\tau^2\sigma^2\right), \quad [3.4]$$

$$\lambda_j^2 \sim \pi\left(\lambda_j^2\right) d\lambda_j^2, \quad [3.5]$$

$$\tau^2 \sim \pi\left(\tau^2\right) d\tau^2. \quad [3.6]$$

Note that conditioning on the scale parameter $\sigma^2 > 0$ is important to guarantee a unique posterior distribution (see Park and Casella 2008). The hyperparameters $\tau^2 > 0$ and $\lambda_1^2, \ldots, \lambda_p^2$ determine the type of sparsity that is enforced on the regression coefficients. In particular, τ^2 is the *global* shrinkage parameter that controls the amount of overall shrinkage for the slope parameters $\boldsymbol{\beta}$. On the other hand, the hyperparameters $\lambda_1^2, \ldots, \lambda_p^2$ correspond to the *local* shrinkage components that determine the amount of shrinkage that is applied to each regression coefficient separately.

Interestingly, the formulation of penalized regressions in terms of scale mixtures of normal not only allows for a very efficient computation, but also makes vivid the theoretical differences and similarities between methods that employ a different penalty term.

3.4.1. *Shrinkage priors*

By adopting a Bayesian approach, the penalization parameters are treated as random variables with their own prior distributions. In this section we provide specific forms of the hierarchical priors outlined in equations [3.5]–[3.6] for different types of penalization. We use the term *shrinkage priors* to emphasize that these prior distributions aim to shrink small effects towards zero. Next, we describe each prior in turn. Throughout, in the absence of prior expert knowledge, we assume a hierarchical prior for the

global shrinkage parameter τ^2 (see, for example, Makalic and Schmidt 2015) such that,

$$\tau^2|\xi \sim IG\left(1/2, 1/\xi\right), \qquad \xi \sim IG\left(1/2, 1\right),$$

where $\xi > 0$ is the mixing parameter and $IG\left(a, b\right)$ is the inverse-Gamma distribution with shape and scale parameters a and b, respectively[9].

3.4.1.1. *Ridge*

The ridge prior corresponds to a normal mixture prior with only a global shrinkage parameter τ^2, i.e. $\lambda_j = 1$ for $j = 1, \ldots, p$, which implies that no regressor-specific local shrinkage is considered (see, for example, Hsiang 1975), i.e.

$$\beta_j|\tau^2, \sigma^2 \sim N\left(0, \tau^2\sigma^2\right). \qquad [3.7]$$

In practice, the ridge regression shrinks the beta coefficients by imposing a penalty on their size[10]. By imposing a size constraint, ridge regressions alleviate the concern that predictors may be highly correlated in the linear regression model. In fact, with highly correlated predictors, the standard OLS coefficients can be poorly determined, as a large positive beta on a given variable can be offset by a similarly large negative beta on a correlated pair.

3.4.1.2. *Lasso*

The lasso is a shrinkage regression method like ridge, but with important differences. Unlike ridge, the nature of the penalization term shrinks those coefficients sufficiently to be exactly zero. In this respect, while ridge is a dense model with a closed-form solution, the lasso is a sparse model in which there is no closed-form solution (see Chapter 3, p. 69, Friedman *et al*. 2001, for a comprehensive discussion). The standard lasso penalty term is an L_1 penalization on the regression coefficients.

When proposing the lasso algorithm, Tibshirani (1996) notices that the lasso penalty is equivalent to the posterior mode estimate under a Laplace

9 We can show that this mixture of inverse-Gamma distribution represents an equivalent specification of the half-Cauchy distribution with mean zero and a scale parameter equal to one. Gelman *et al*. (2006), Polson and Scott (2010) and Polson *et al*. (2012) recommend the half-Cauchy distribution as a sensible prior specification for τ.

10 Ridge regressions are not scale-invariant, so it is common to standardize the predictors.

prior. Park and Casella (2008) build on this intuition and suggest that a consistent prior can be defined as a scale mixture of normals with an exponential mixing density, i.e.

$$\begin{aligned}
\boldsymbol{\beta}|\sigma^2, \boldsymbol{D}_\lambda &\sim N\left(\mathbf{0}, \sigma^2 \boldsymbol{D}_\lambda\right), \qquad \boldsymbol{D}_\lambda = \text{diag}\left(\lambda_1^2, \ldots, \lambda_p^2\right) \\
\lambda_1^2, \ldots, \lambda_p^2|\tau^2 &\sim \prod_{j=1}^{p} \frac{\tau^2}{2} \exp\left(-\tau^2 \lambda_j^2/2\right) d\lambda_j^2, \qquad \lambda_1, \ldots, \lambda_p > 0 \\
\sigma^2 &\sim \pi\left(\sigma^2\right) d\sigma^2, \qquad \sigma^2 > 0
\end{aligned} \tag{3.8}$$

With such a prior, the posterior mode estimates are similar to the estimates under the L_1 penalty term[11]. The key advantage of the Bayesian lasso, with respect to a frequentist approach, is that the penalization is done via both a global shrinkage parameter τ^2 and a local, i.e., regressor-specific, shrinkage parameter. Hence, the Bayesian lasso expresses the penalized regression as a global–local shrinkage estimator. This, in turn, increases the flexibility of the model.

3.4.1.3. *Elastic net*

The popularity of the lasso is due to its ability to perform clear-cut model selection by shrinking betas to exactly zero, unlike the ridge prior, which has more of a regularization effect. However, the lasso, in its standard form, is not immune to flaws. First, it cannot deal with a framework where $p > T$ after selection, meaning it cannot deal with a situation where the selected regressors are more than the available observables. Second, when a group of predictors are highly correlated, the lasso often selects only a few of the variables in the group, despite the true predictive content of the other variables. Third, Polson and Scott (2010) show that the lasso penalty can lead to over-shrinkage, namely, it does not enjoy the so-called "oracle property", i.e. the model selected by the lasso is not necessarily the true data-generating process. In this respect, Fan and Li (2001) and Zou (2006) show that the oracle property for the lasso is satisfied only based on some stringent conditions. These flaws motivated the development of variants of the simple lasso, such as the elastic net.

11 Note that by integrating λ_j^2 out, we obtain the original Tibshirani (1996) double-exponential prior.

Originally proposed by Zou and Hastie (2005), the elastic net mitigates the lasso concerns related to $p > T$ and the presence of group-wise correlated regressors. Li *et al*. (2010) show that the shrinkage prior for the elastic net penalization can be expressed as

$$\pi\left(\boldsymbol{\beta}|\sigma^2\right) \propto \exp\left(-\frac{\tau_1^2}{\sigma}\sum_{j=1}^{p}|\beta_j| - \frac{\tau_2^2}{2\sigma}\sum_{j=1}^{p}\beta_j^2\right). \qquad [3.9]$$

This prior can be rewritten in a hierarchical form as follows:

$$\begin{aligned}
&\boldsymbol{\beta}|\sigma^2, \boldsymbol{D}_\lambda^* \sim N\left(\mathbf{0}, \sigma^2\boldsymbol{D}_\lambda^*\right), \\
&\lambda_1^2, \ldots, \lambda_p^2|\tau_1^2 \sim \prod_{j=1}^{p}\frac{\tau_1^2}{2}\exp\left(-\tau_1^2\lambda_j^2/2\right)d\lambda_j^2, \qquad \lambda_1, \ldots, \lambda_p > 0
\end{aligned} \qquad [3.10]$$

where $\boldsymbol{D}_\lambda^*$ is a diagonal matrix with elements $\left(\lambda_j^{-2} + \tau_2^2\right)^{-1}, j = 1, \ldots, p$. Note that in this case, $\boldsymbol{\beta}$ is not conditionally independent on τ_2^2 as it appears in the covariance matrix. However, this term enters into the exponent of the posterior as a "Gaussian" component so that the Gibbs sampler continues to be straightforward (see Appendix section 3.8.).

Li *et al*. (2010) show that the corresponding posterior mode estimates are equivalent to the estimates obtained from the classical elastic net penalty. More specifically, the two parameters τ_1^2, τ_2^2 determine the relative effects of the lasso and the ridge penalty, respectively. The advantage of the Bayesian approach is that the parameters τ_1^2, τ_2^2 can be estimated jointly, while in the standard frequentist approach one needs to implement a sequential cross-validation procedure. However, the latter has been shown to result in over-shrinkage of the coefficients (see Zou and Hastie 2005).

3.4.1.4. *Group-lasso*

Yuan and Lin (2006) extended the standard lasso penalized regression by explicitly taking into account the fact that predictors can be highly correlated and even grouped in some way. This is the case of macroeconomic variables and/or firms' characteristics (see, for example, Bianchi and McAlinn 2018

and Bianchi *et al.* 2019a). Kyung *et al.* (2010) show that the group-lasso corresponds to a conditional prior of the form

$$p\left(\beta_j|\sigma^2,\tau^2\right) = C\exp\left\{-\frac{\tau^2}{\sqrt{\sigma^2}}\sum_{g=1}^{G}\|\boldsymbol{\beta}_g\|\right\}, \quad \text{for} \quad g=1,\ldots,G \tag{3.11}$$

where G is the number of groups, $\boldsymbol{\beta}_g$ is the set of predictors for group g, $\|\boldsymbol{\beta}_g\| = \left(\boldsymbol{\beta}_g'\boldsymbol{\beta}_g\right)^{1/2}$ and C denotes a normalizing constant. The grouping effect leads to a different hierarchical specification, i.e. the regression coefficients cannot be considered independent within groups. As a result, Kyung *et al.* (2010) proposed the following hierarchical multivariate model:

$$\begin{aligned}\boldsymbol{\beta}_g|\lambda_g^2,\sigma^2 &\sim MN\left(\mathbf{0},\sigma^2\lambda_g^2\mathrm{I}_{m_g}\right),\\ \lambda_g^2|\tau^2 &\sim Gamma\left(\frac{m_g+1}{2},\frac{\tau^2}{2}\right), \quad \text{for} \quad g=1,\ldots,G\end{aligned} \tag{3.12}$$

where $MN(a,A)$ denotes a multivariate normal distribution with mean vector a and covariance matrix A. Here, m_g and I_{m_g} denote the dimension of the group g and an $(m_g \times m_g)$ identity matrix, respectively. It is easy to see that a large value of τ^2 implies that all coefficients in a given group will be shrunk simultaneously to zero.

3.4.1.5. *Horseshoe and group horseshoe*

Polson and Scott (2010) showed that a good prior specification for the local shrinkage hyperparameters should (1) be centered at zero to ensure efficiency in recovering the true sampling distribution of the regression parameters and (2) have polynomial tails to shrink noisy variables more aggressively towards zero. However, neither the ridge nor the lasso regression or their generalizations, such as elastic-net and group-lasso, can mitigate the issue of having large outlying signals and sparsity of unknown form.

To solve these issues Carvalho *et al.* (2009 and 2010) propose a fully Bayesian framework for sparse supervised-learning based on the so-called horseshoe prior. Such a prior is a member of the scale mixture of normals, and is therefore closely related to existing approaches, such as the lasso, the ridge and the elastic net penalties.

Makalic and Schmidt (2015) showed that, by introducing latent auxiliary parameters ν_j, ξ, the original horseshoe regression can be rewritten as a hierarchical model as follows:

$$\beta_j|\lambda_j^2, \tau^2, \sigma^2 \sim N\left(0, \lambda_j^2\tau^2\sigma^2\right) \quad [3.13]$$

$$\lambda_j^2|\nu_j \sim IG\left(1/2, 1/\nu_j\right), \quad [3.14]$$

$$\tau^2|\xi \sim IG\left(1/2, 1/\xi\right), \quad [3.15]$$

$$\nu_1, \ldots, \nu_p, \xi \sim IG\left(1/2/, 1\right) \ . \quad [3.16]$$

Such a hierarchical model makes the sampling from the posterior distribution quite simple (see Appendix section 3.8.)[12].

Importantly, the horseshoe prior is the only shrinkage prior which has an asymptote at both zero and fat tails (see Carvalho *et al*. 2010). These two properties combined make sure that small coefficients are heavily shrunk towards zero, while large coefficients remain large. This is different from the lasso and elastic net, whereby both small and large coefficients are regularized and shrunk towards smaller values.

The horseshoe prior has the ability to adapt to different sparsity patterns, while simultaneously avoiding over-shrinkage of large coefficients. However, similar to other approaches, the horseshoe prior is based on the assumption that all the regression coefficients β_j are conditionally independent, i.e. no group structure is allowed for the predictors. Xu *et al*. (2016) relax this assumption and propose a grouped horseshoe prior as follows:

$$\begin{aligned}
\beta_j|\lambda_g^2, \tau^2, \sigma^2 &\sim N\left(0, \lambda_g^2, \tau^2\sigma^2\right) \\
\lambda_g^2|\nu_g &\sim IG\left(1/2, 1/\nu_g\right), \\
\tau^2|\xi &\sim IG\left(1/2, 1/\xi\right), \\
\nu_1, \ldots, \nu_G, \xi &\sim IG\left(1/2/, 1\right) \qquad [3.17]
\end{aligned}$$

12 The original horseshoe prior implies that the local shrinkage parameters are sampled from a half-Cauchy distribution, i.e., $\lambda_j \sim C^+\left(0, 1\right)$. Such a distribution can equivalently be written as a mixture of inverse Gamma distributions, i.e.,

$$\lambda_j^2|\nu_j \sim IG\left(1/2, 1/\nu_j\right), \qquad \nu_j \sim IG\left(1/2, 1\right) \ .$$

where λ_g^2 denotes the shrinkage parameter at the group level g and τ^2 denotes the global shrinkage parameter. Similar to the group-lasso, the group-horseshoe shrinks all variables within a group to zero or leaves them unshrunk.

3.4.2. *Forecast evaluations*

Following standard practice in the forecasting literature, we evaluate the quality of the predictive strategy from competing models, based on both point and density forecasts. In particular, we first compare predictive strategies based on the out-of-sample mean squared predictive error (RMSE), i.e.

$$MSPE = \frac{1}{T-\tau}\sum_{t=\tau}^{T-1}\left(y_{t+1} - \hat{y}_{t+1}\right)^2 ,$$

where $T - \tau$ represents the out-of-sample period for the horizon, $\hat{y}_{t+1} = E\left[y_{t+1}|\mathcal{M}_s\right]$ is the one-step ahead point forecast conditional on information up to time t from the sparse predictive strategy $\mathcal{M}_s$ and y_{t+1} is the realized portfolio returns.

In addition to the MSPE, we compute the predictive R^2_{oos}, as suggested by Campbell and Thompson (2007), to compare the forecasts from each model, i.e. $\hat{y}_{t+1}$, to the forecast obtained by the simple unconditional mean of the portfolio returns, which we denote by $\overline{y}_{t+1}$, i.e.

$$R^2_{oos} = 1 - \frac{\sum_{t=\tau}^{T-1}\left(y_{t+1} - \hat{y}_{t+1}\right)^2}{\sum_{t=\tau}^{T-1}\left(y_{t+1} - \overline{y}_{t+1}\right)^2} . \qquad [3.18]$$

It can be easily seen that R^2_{oos} is akin to the familiar in-sample R^2 statistics and measures the reduction in the prediction error obtained from a given sparse predictive model relative to the recursive mean. Thus, when $R^2_{oos} > 0$, the $\hat{y}_{t+1}$ forecast outperforms the historical average forecast in a mean squared error sense.

Although informative, performance measures based on point forecasts only give a partial assessment. Ideally, we also want to compare the predictive densities across strategies. In fact, performance measures based on predictive densities weigh and compare dispersion of forecasts along with their location,

as well as elaborate on raw MSPE measures. That is, performance measures based on the predictive density provide an assessment of a model's ability to explain not only the expected value, i.e. the risk premium, but also the overall distribution of the portfolio excess returns, naturally penalizing the size/complexity of different models. Hence, comparing both point and density forecasts provides a broader understanding of the predictive abilities of the different strategies. Specifically, we compare penalized models (and their implied strategies) based on the log predictive density ratios (LPDR), i.e.

$$\text{LPDR} = \sum_{t=\tau}^{T-1} \log\{p(y_{t+1}|\mathcal{M}_s)/p(y_{t+1}|\mathcal{M}_0)\}, \quad [3.19]$$

where $p(y_{t+1}|\mathcal{M}_s)$ is the predictive density computed at time t under the sparse predictive regression model indexed by $\mathcal{M}_s$. We compare such a model against a benchmark forecasting framework denoted by $\mathcal{M}_0$. As a benchmark we used the ridge regression which is the only pure "dense" approach in our set of models (see Giannone *et al*. 2018). The logic of using a ridge shrinkage prior as a benchmark is two-fold; first, it allows us to assess the actual benefit of sparsity – meaning shrinking betas to zero – for out-of-sample predictability, i.e., the variance-bias tradeoff. Second, it allows us to define a pecking order (within our context of portfolio excess returns) of sparse predictive approaches with respect to a dense modeling framework commonly used in large-dimensional linear regressions.

3.5. Predicting asset returns: empirical results

3.5.1. *Statistical performance*

We estimate each sparse predictive model on a rolling window basis. In particular, at each time t, a prediction is generated based on the previous $\tau = 120$ monthly observations of portfolio returns and predictors. Forecasts are fully out-of-sample, meaning that forecasts at time t for y_{t+1} are based on parameter estimates obtained from the lagged predictors $\boldsymbol{x}_{t-120:t-1}$ and portfolio returns $y_{t-119:t}$ according to the predictive linear relationship in equation [3.1]. As a result, forecasts are compared with realized returns that have not been used to estimate the model parameters.

Table 3.1 shows the performance results across different portfolios. For the aggregate stock market portfolio, the MSPE is quite comparable across regression models, with a slightly better performance obtained by the group-horseshoe prior. The group-horseshoe also slightly outperforms the competing methodologies in the case of low-value and past-winner portfolios. Interestingly, a dense model, like the ridge regression, compares favorably against pure sparse regression methods, like the lasso. In fact, lasso and ridge deliver performances that, in a mean squared error sense, are almost indistinguishable. As a whole, there is no clear pecking order among methodologies when we look at the MSPE per se.

Things become a bit clearer when we look at the predictive R^2_{oos}. The elastic net is the only methodology that delivers a positive predictive R^2_{oos} throughout portfolios. Interestingly, except few nuances (see, for example, Feng *et al*. 2018), the elastic net is a methodology that has been overlooked so far in the empirical finance literature. Elastic net produces a regression model that is penalized with both the L1-norm and L2-norm. The consequence of this is to effectively shrink coefficients (like in ridge regression) and to set some coefficients to zero (as in lasso). As a result, it is a natural candidate to exploit the benefit of both model selection and parameter regularization. The results in terms of predictive R^2_{oos} suggest that choosing between model selection and regularization may not be an optimal strategy for linear regressions in a data-rich environment.

Table 3.1 also reports the log-predictive density ratios (LPDR), comparing each of the sparse predictive regression models against the ridge regression. As explained above, the ridge regression is used as a benchmark, as it is the only pure dense modeling framework we consider. By using the ridge regression as a benchmark, we can disentangle the benefits of imposing sparsity in the data-generating process in addition to, or separate from, simply regularizing the estimates of the regression betas. A positive LPDR is evidence in favor of the alternative predictive strategy against the ridge regression.

Note that performance measures based on predictive densities weigh and compare dispersion of forecast densities along with location, and elaborate on raw RMSE measures; comparing both measurements, i.e., point and density forecasts, gives a broader understanding of the predictive abilities of the different strategies.

Portfolio	Ridge			Lasso			Horseshoe			Group-lasso			Elastic net			Group-horseshoe		
	MSPE	R^2_{oos}	LPDR	MSPE	R^2_{oos}	LPDR	MSPE	R^2_{oos}	LPDR	MSPE	R^2_{oos}	LPDR	MSPE	R^2_{oos}	LPDR	MSPE	R^2_{oos}	LPDR
Stock market	20.89	0.007	–	20.88	0.008	18.38	20.80	0.012	–110.26	20.80	0.012	–68.24	21.02	0.001	–825.34	20.77	0.013	–129.27
Equity high value	25.63	–0.016	–	25.65	–0.016	34.98	25.52	–0.011	–207.26	25.36	–0.005	3.21	25.21	0.001	–1174.9	25.62	–0.015	–180.57
Equity low value	33.78	0.012	–	33.66	0.016	–7.21	33.22	0.029	–135.91	33.64	0.017	–37.28	34.15	0.002	–1767.52	33.34	0.025	–225.14
Equity past winner	41.15	0.005	–	41.15	0.005	–13.35	41.01	0.009	–139.73	41.12	0.006	–99.08	41.32	0.001	–1959.63	40.96	0.010	–227.66
Equity past loser	42.59	0.135	–	49.16	0.001	56.51	50.41	–0.024	–5.31	48.86	0.007	94.04	49.17	0.001	–2293.43	49.80	–0.012	–40.99
FX high value	5.99	0.020	–	5.99	0.021	22.73	6.01	0.018	–37.61	5.98	0.021	–76.37	6.11	0.001	668.84	6.01	0.017	–67.91
FX low value	6.86	–0.017	–	6.86	–0.016	25.57	6.91	–0.025	–82.88	6.79	–0.007	0.60	6.74	0.001	507.57	6.98	–0.035	–186.29
FX past winner	6.49	–0.007	–	6.50	–0.007	–53.03	6.53	–0.012	–142.87	6.45	0.000	–35.71	6.45	0.001	438.88	6.55	–0.015	–200.98
FX past loser	6.25	0.018	–	6.26	0.017	20.81	6.34	0.005	–43.71	6.23	0.022	–102.82	6.36	0.002	519.69	6.34	0.004	–108.99
Sovereign high beta	6.53	–0.124	–	6.58	–0.132	–15.13	6.60	–0.136	–44.41	6.33	–0.089	3.91	5.80	0.002	101.87	6.68	–0.150	–130.50
Sovereign low beta	4.49	–0.341	–	4.59	–0.369	4.58	4.71	–0.405	–40.61	4.41	–0.315	6.72	3.35	0.001	146.34	4.71	–0.405	–35.65
Treasury	0.47	0.697	–	0.48	0.690	–27.15	0.59	0.620	–597.27	0.48	0.686	1.55	1.52	0.012	164.47	0.58	0.623	–621.13
Corporate	3.13	–0.053	–	3.14	–0.058	–1.65	3.13	–0.054	–155.89	3.07	–0.033	6.50	2.97	0.001	636.52	3.14	–0.056	–255.11

This table reports the out-of-sample comparison of the sparse predictive regression models across different portfolios and asset classes. The methodologies reported are ridge, lasso, horseshoe, group-lasso, elastic net, and group-horseshoe. For each methodology, we report the mean squared prediction error (MSPE), the predictive R^2_{oos} and the log-predictive density ratio (LPDR).

Table 3.1. *Out-of-sample statistical performance*

The empirical evidence suggests that sparsity does help to improve the out-of-sample performance of linear predictive regressions. The lasso, the group-lasso and the elastic net tend to outperform the ridge regression on average across portfolios. When we focus on the alternative asset classes, we observe that the elastic net delivers the most consistent and strongest outperformance. Interestingly, at least in the context of our application, the horseshoe shrinkage prior and its group-based extension do not deliver a good performance. As a whole, the statistical evidence points in favor of the benefit of sparsity for linear predictive regressions.

Delving further into the performance evaluation of different sparse predictive regressions, in the next section we will implement a simple rotation strategy separately for each portfolio and take the point of view of an investor who needs to allocate wealth between a given portfolio and a riskless asset.

3.5.2. *Economic significance*

The amount of time variation in expected excess returns that is detectable by sparse predictive regressions naturally calls for successful market-timing strategies. Following the intuition in Gallant *et al*. (1990), an investor who moves into the market in times of higher Sharpe ratios, and who, conversely, moves our at times of lower Sharpe ratios, would obtain a higher performance than a buy-and-hold investor. Thus, along with the statistical performance of each sparse regression strategy, we measure the potential benefit of market-timing based on return predictability.

In particular, we follow Marquering and Verbeek (2004), Campbell and Thompson (2007), Welch and Goyal (2007), Rapach *et al*. (2009), Wachter and Warusawitharana (2009) and Rapach and Zhou (2013), and calculate the realized utility gains for a mean-variance investor who wants to allocate their wealth between a given portfolio and a risk-free asset. More specifically, we compute the optimal allocation to the risky asset $\omega_{j,t}$ obtained from the forecast of a given sparse predictive regression, i.e.

$$\omega_{j,t} = \left(\frac{1}{\gamma}\right)\left(\frac{\hat{y}_{j,t+1}}{\hat{\sigma}^2_{t+1}}\right), \qquad [3.20]$$

where $\hat{y}_{j,t+1}$ is the one-step ahead forecast from the j-th predictive model, γ is the investor's risk aversion, and $\hat{\sigma}^2_{t+1}$ is the rolling-window estimate of the variance of the portfolio returns[13]. The investor who implements the optimal allocation [3.20] realizes an average utility equal to

$$\hat{\nu}_j = \hat{\mu}_j - \frac{1}{2}\gamma\hat{\sigma}^2_j,$$

where $\hat{\mu}_j$ and $\hat{\sigma}^2_j$ are the sample mean and variance of the portfolio excess returns computed over the out-of-sample period. In our applications, we compare the realized utility from the given forecasting model to the one obtained from a simple rolling-window estimate of the average portfolio return $\overline{y}_t$. The utility gain is computed as the difference between the realized utility obtained from the optimal allocation [3.20] and its counterpart obtained by simply replacing $\hat{y}_{j,t+1}$ with $\overline{y}_{j,t+1}$. Economically, the utility gain can be interpreted as the fee a mean-variance investor would be willing to pay to have access to the predictive content of the sparse regression model on a given portfolio, relative to just relying on the historical mean alone.

Table 3.2 reports the results for both a moderately low level of risk aversion equal to $\gamma = 5$ (panel A) and a higher level of risk aversion equal to $\gamma = 10$ (panel B). The results are qualitatively similar for other reasonably lower and/or higher levels of risk aversion.

The first column reports the results from a dense ridge regression model. Excepting a few nuances, ridge regressions tend to generate a negative utility gain, i.e. an investor would be better off using a simple rolling-window estimate of the conditional mean as a trading signal than using the forecasts from a regularized predictive regression. Predictability becomes more valuable when model selection, i.e. sparsity, is explicitly acknowledged. The elastic net penalty produces a positive utility gain for almost all portfolios. In particular, a representative investor with $\gamma = 5$ is willing to pay approximately 1% on an annual basis to access the information produced by

13 The conclusions are robust to using an estimate of volatility based on the exponential weighted moving average (EWMA). We view the statistical and economic analysis of more complex, machine learning methodologies to forecast the second, rather than the first, moment as an interesting avenue for future research.

the sparse predictive regression with both an L1-norm and an L2-norm penalty. Note there is a substantial variation in the performance of the elastic net across portfolios. Take $\gamma = 5$ as an example; an investor is willing to pay up to 6% annually for the sovereign high-beta strategy, but only 0.12% annually for corporate bonds. The predictability of value strategies is also more valuable in economic terms relative to that found for momentum strategies: an investor would pay a positive fee of about 1% (0.5%) for high-value equity (currency) portfolios to access the information produced by the elastic net.

	Panel A: Risk aversion equal to $\gamma = 5$					
Portfolio	Ridge	Lasso	Horseshoe	G-lasso	Elastic net	G-Horseshoe
Stock market	–1.460	–1.493	–0.864	–0.875	0.005	–0.853
Equity high value	–3.758	–4.440	–3.563	–2.425	0.075	–3.629
Equity low value	–0.960	–1.142	–0.877	–0.537	0.056	–1.198
Equity past winner	–1.972	–1.363	–1.747	–0.952	–0.024	–1.878
Equity past loser	–0.492	–18.524	–32.145	–8.849	–0.024	–28.24
FX high value	0.028	0.053	0.326	0.142	0.036	0.143
FX low value	–1.270	–1.142	–0.739	–0.690	0.000	–0.865
FX past winner	–1.700	–1.642	–1.185	–1.207	–0.004	–1.593
FX past loser	0.413	0.497	0.268	0.364	0.040	0.421
Sovereign high beta	–29.47	–30.97	–30.48	–27.64	0.52	–31.98
Sovereign low beta	–5.91	–6.25	–7.81	–5.66	0.08	–7.68
Treasury	–2.79	–3.74	–5.20	–3.54	0.13	–5.31
Corporate	–1.28	–1.75	–5.84	–0.92	0.01	–5.93

	Panel B: Risk aversion equal to $\gamma = 10$					
Portfolio	Ridge	Lasso	Horseshoe	G-lasso	Elastic net	G-Horseshoe
Stock market	–0.748	–0.783	–0.298	–0.509	–0.009	–0.311
Equity high value	–1.662	–1.982	–1.418	–1.150	0.043	–1.606
Equity low value	–0.091	–0.214	0.166	–0.045	0.028	–0.130
Equity past winner	–0.988	–0.697	–0.984	–0.558	–0.011	–1.042
Equity past loser	0.884	–6.662	–13.177	–2.738	–0.008	–11.33
FX high value	0.223	0.272	0.475	0.261	0.028	0.315
FX low value	–0.491	–0.398	–0.120	–0.191	0.002	–0.213
FX past winner	–0.871	–0.800	–0.439	–0.596	–0.002	–0.670
FX past loser	0.522	0.624	0.714	0.409	0.033	0.747
Sovereign high beta	–13.21	–13.96	–13.77	–12.37	0.28	–14.55
Sovereign low beta	–2.50	–2.65	–3.44	–2.38	0.04	–3.39
Treasury	–1.49	–1.97	–2.68	–1.84	0.10	–2.79
Corporate	0.67	0.53	–1.14	0.73	0.02	–1.19

This table reports the out-of-sample average utility gain for each of the sparse predictive strategies and for each portfolio. The benchmark is a strategy based on the conditional mean. We report the results for both a risk aversion equal to $\gamma = 5$ (top panel) and equal to $\gamma = 10$ (bottom panel).

Table 3.2. *Out-of-sample economic significance*

As a whole, Table 3.2 confirms the performance of the elastic net penalty when it comes to actual economic significance. Other predictive strategies outperform the trading signal from a simple rolling-window mean estimate, however, the evidence is much less systematic. For instance, while the horseshoe prior delivers some positive utility gain for high-value and past-loser portfolios in currencies (with annual fees higher than 3%), it does

not deliver significant gains for other asset classes. A similar comment can be made for the lasso, the group-lasso and the group-horseshoe.

3.6. Discussion on the dynamics of sparsity

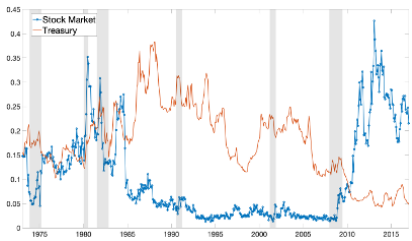
Regression models that include some form of time variation in the betas have become highly popular in the financial economics literature. Relevant papers include Pesaran and Timmermann (2002), Lewellen (2004), Avramov (2004), Jostova and Philipov (2005), Welch and Goyal (2007), Adrian and Franzoni (2009), Rapach *et al*. (2009), Dangl and Halling (2012), Pettenuzzo *et al*. (2014), Bianchi and McAlinn (2018), Kelly *et al*. (2018) and Bianchi *et al*. (2019b), among others.

In practice, however, the empirical application of time-varying parameter models is often restricted to a handful of predictors, either justified by existing economic theories or previous empirical findings. In light of the evidence that information in many predictors can be beneficial both from a statistical and an economic perspective, the inability of existing time-varying parameter methodologies to be used in high-dimensional settings seems to be a fundamental shortcoming. A key issue in modeling time-varying parameter models in a large dimensional setting is the possibility of accounting for dynamic sparsity. In fact, it is not necessarily true *a priori* that the relevant predictors are always the same through time. For instance, Bianchi *et al*. (2019b) show that the level of sparsity in the cross-section of expected returns can be highly time-varying and related to aggregate financial conditions.

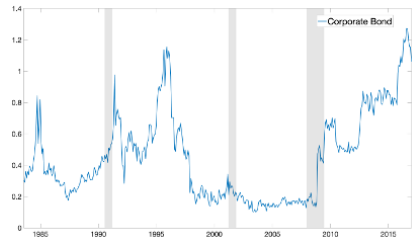
Related to that, in this section we provide further evidence that, indeed, sparsity is not necessarily constant over time, but is quite heterogeneous both in the time series and in the cross section. In particular, as a case in point, we report the posterior mean estimates $\hat{\tau}^2$ from the lasso penalized regression. The hierarchical nature of the lasso shrinkage prior allows us to interpret τ^2 as the overall shrinkage parameter while keeping the flexibility of having local, i.e. predictor-specific, shrinkage parameters $\lambda_j^2|\tau^2, j = 1, \ldots, p$ as well.

Figure 3.1 reports the results for each asset class. The top left panel shows the rolling window estimates $\hat{\tau}^2$ for the aggregate stock market and the treasury portfolio. The time variation in shrinkage for the stock market is of particular interest. Sparsity seems to be relatively low during a recession, i.e. the amount of information that is relevant to predict the aggregate stock market tends to be

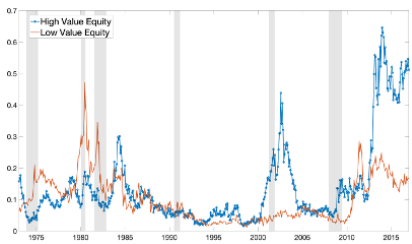
larger during recessions versus expansions. In fact, the last part of the sample suggests that sparsity has become more prominent in the aftermath of the great financial crisis of 2008/2009.



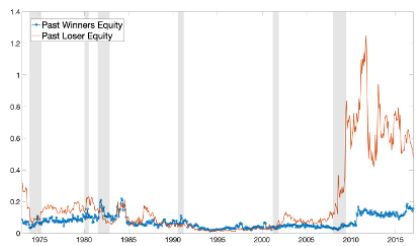
(a) Stock Market and Treasury



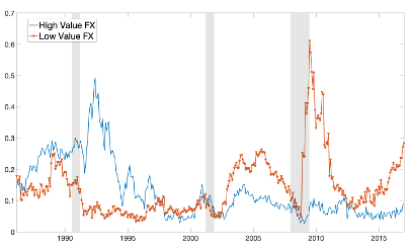
(b) Corporate Bonds



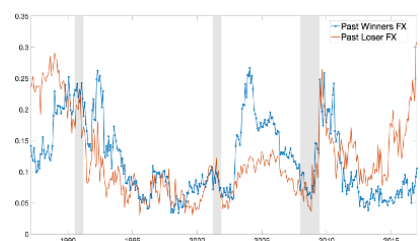
(c) Value Equity



(d) Momentum Equity



(e) Value FX



(f) Momentum FX

This figure shows the posterior mean of the aggregate sparsity parameter τ^2 obtained from the lasso as shown in section 3.4.1.2. The model is recursively estimated on a rolling window basis by using 120 observations as window size. A full description of the data is in section 3.3.

Figure 3.1. *Sparsity across portfolios. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Similar dynamics can be observed for value equity portfolios (mid-left panel): we can easily see that sparsity tends to increase before and after the great financial crisis, in particular for the high book-to-market portfolio. The

model-implied conditional expectation of high-value stock excess returns is particularly sparse (as suggested by a higher $\hat{\tau}^2$) toward the end of our sample. This is also the case for corporate bonds (top right panel) and past-losers (mid-right panel). The only asset class for which we find a relatively stable and low sparsity is the past-winner portfolio.

Interestingly, past-winner and past-loser FX strategies (bottom-right panel) seem to share similar dynamics for sparsity in model-implied expected excess returns featuring large and long-lasting waves. Instead, when we look at value strategies in currencies (bottom-left panel), the dynamics of sparsity between high- and low-value portfolios differ over time, with the "spread" between τ^2s in the two portfolios that increases before and after the great financial crisis.

One comment is in order. A full-blown investigation of time-varying sparsity is beyond the scope of this chapter. However, we believe that our results show some preliminary evidence that the dynamics of sparsity likely have relevant implications for both forecasting and the modeling of expected returns in a linear regression setting, and therefore should be taken seriously from a methodological point of view.

3.7. Conclusion

In this chapter, we evaluate a large set of penalized predictive regression models for the prediction of excess returns across different asset classes within a data-rich environment. Empirically, we make use of asset-specific predictors as well as a large set of economic variables from the FRED-MD database, which arguably capture several potentially relevant risks in the macroeconomy. Methodologically, we adopt a Bayesian estimation framework in which the prior distribution performs a function similar to that of the penalty term in classical penalization. We explore both the econometric underpinnings of each methodology and their economic gains across different asset classes. The main results show that the penalty term that implies both shrinkage on the model space – by setting some regression coefficients to zero – and regularization of the remaining regression coefficients tends to outperform both dense (e.g. ridge regression) and sparse (e.g. lasso regression) methodologies. Finally, we show that the amount of shrinkage, or the amount of sparsity, tends to evolve over time and differ for each asset class. This means assuming that sparsity is

constant in the data-generating process may be costly in terms of out-of-sample performance, both statistically and economically.

3.8. Appendix

Bayesian inference for a statistical model requires the simulate from the posterior distributions of all model parameters. This appendix provides details regarding the posterior simulation of parameters of interest for each of the penalized regression methodologies used in this chapter.

3.9. Posterior simulation

3.9.1. *Ridge regression*

The ridge regression prior [3.7] implies a closed-form penalized least squares representation of the form

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}'\boldsymbol{X} + \frac{1}{\tau^2}\mathrm{I}_p \right)^{-1} \boldsymbol{X}'\boldsymbol{y}$$

where $\boldsymbol{X}$ and $\boldsymbol{y}$ are the matrix of predictors and the vector of response variable, respectively. The posterior mean estimates correspond to the estimates obtained by using a standard L_2 penalty term, i.e., $\phi(\boldsymbol{\beta};\cdot) = \tau^{-2}\sum_{j=1}^{p}\beta_j^2$. The parameter τ determines the amount of global shrinkage, with smaller values resulting in more shrinkage and with $\tau \to \infty$ obtaining the OLS estimator.

3.9.2. *Lasso and group-lasso*

Park and Casella (2008) show that the hierachical specification in equation [3.8] translates in a relatively standard Gibbs sampler, which exploits the conjugacy of the inverse Gaussian distribution. More precisely, the conditional distribution of the regression parameters is given by

$$\boldsymbol{\beta}|\sigma^2, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim N\left(\boldsymbol{A}^{-1}\boldsymbol{X}'\tilde{\boldsymbol{y}}, \sigma^2\boldsymbol{A}^{-1}\right) \qquad [3.21]$$

with $\boldsymbol{A} = \left(\boldsymbol{X}'\boldsymbol{X} + \boldsymbol{D}_{\lambda}^{-1}\right)$. Similarly, the posterior distribution for σ^2 is conjugate and takes the form of a conditional inverse gamma, i.e.

$$\sigma^2|\boldsymbol{\beta}, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim IG\left(\frac{n-1+p}{2}, \frac{1}{2}\left(\tilde{\boldsymbol{y}} - \boldsymbol{X}\boldsymbol{\beta}\right)'\left(\tilde{\boldsymbol{y}} - \boldsymbol{X}\boldsymbol{\beta}\right) + \frac{\tau}{2}\boldsymbol{\beta}'\boldsymbol{D}_{\lambda}^{-1}\boldsymbol{\beta}\right). \quad [3.22]$$

Finally, the conditional posterior distribution of the local shrinkage parameter $1/\lambda_j^2$ is the inverse Gaussian distribution with local and scale parameters as follows:

$$1/\lambda_j^2 = \gamma_j|\boldsymbol{\beta}, \sigma^2, \boldsymbol{X}, \boldsymbol{y} \sim \text{Inverse-Gaussian}\left(\frac{\tau^2\sigma^2}{|\beta_j|}, \tau^2\right)\mathbb{I}\left(\gamma_j > 0\right). \quad [3.23]$$

As far as the group-lasso is concerned, Kyung *et al.* (2010) show that the full conditional distribution of the parameters for the group-lasso is still available in closed form. For instance, the posterior distribution of the regression parameters is a multivariate normal, i.e.

$$\boldsymbol{\beta}_{G_k}|\boldsymbol{\beta}_{-G_k}, \sigma^2, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim N\left(\boldsymbol{B}_k^{-1}\boldsymbol{X}_k'\left(\tilde{\boldsymbol{y}} - \frac{1}{2}\sum_{k' \neq k} X_{k'}\boldsymbol{\beta}_{G_{k'}}\right), \sigma^2 B_k^{-1}\right), \quad [3.24]$$

with $\boldsymbol{\beta}_{-G_k} = \left(\boldsymbol{\beta}_{G_1}, \ldots, \boldsymbol{\beta}_{G_{k-1}}, \boldsymbol{\beta}_{G_{k+1}}, \ldots, \boldsymbol{\beta}_{G_K}\right)$, and $B_k = \boldsymbol{X}_k'\boldsymbol{X}_k + \left(1/\lambda_k^2\right)\mathbf{I}_{m_k}$. The full conditional distribution for the local shrinkage parameters and the variance are defined as follows:

$$1/\lambda_k^2 = \gamma_k|\boldsymbol{\beta}, \sigma^2, \boldsymbol{X}, \boldsymbol{y} \sim \text{Inverse-Gaussian}\left(\sqrt{\frac{\tau^2\sigma^2}{\|\beta_{G_k}\|^2}}, \tau^2\right)\mathbb{I}\left(\gamma_k > 0\right)$$

$$\sigma^2|\boldsymbol{\beta}, \lambda_1^2, \ldots, \lambda_K^2, \boldsymbol{X}, \boldsymbol{y} \sim IG\left(\frac{n-1+p}{2}, \frac{1}{2}\sum_{k=1}^{K}\frac{1}{\lambda_k^2}\|\boldsymbol{\beta}_{G_k}\|^2\right). \quad [3.25]$$

Assuming a gamma prior for the global shrinkage parameter $G\left(r, \delta\right)$, the full conditional distribution for the τ^2 is

$$\tau^2|\boldsymbol{\beta}, \sigma^2, \lambda_1^2, \ldots, \lambda_K^2, \boldsymbol{X}, \tilde{\boldsymbol{y}} \sim G\left(\frac{p+K}{2} + r, \frac{1}{2}\sum_{k=1}^{K}\lambda_k^2 + \delta\right).$$

3.9.3. *Elastic net*

The elastic net can be seen as a combination of the ridge regression and the lasso. Li *et al*. (2010) show that the scale mixture of normals prior [3.10] implies a set of full conditional distributions as follows:

$$\begin{aligned}
&\boldsymbol{\beta}|\sigma^2, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim N\left(\boldsymbol{A}^{*-1}\boldsymbol{X}'\tilde{\boldsymbol{y}}, \sigma^2\boldsymbol{A}^{*,-1}\right), \\
&\sigma^2|\boldsymbol{\beta}, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \\
&\quad \sim IG\left(\frac{n-1+p}{2}, \frac{1}{2}\left(\tilde{\boldsymbol{y}}-\boldsymbol{X}\boldsymbol{\beta}\right)'\left(\tilde{\boldsymbol{y}}-\boldsymbol{X}\boldsymbol{\beta}\right)+\frac{\tau}{2}\boldsymbol{\beta}'\boldsymbol{D}_{\lambda}^{*,-1}\boldsymbol{\beta}\right), \\
&1/\lambda_j^2 = \gamma_j|\boldsymbol{\beta}, \sigma^2, \boldsymbol{X}, \boldsymbol{y} \sim \text{Inverse-Gaussian}\left(\frac{\tau_1^2\sigma^2}{\beta_j^2}, \tau_1^2\right)\mathbb{I}\left(\gamma_j > 0\right)
\end{aligned} \tag{3.26}$$

where $\boldsymbol{A}^* = \left(\boldsymbol{X}'\boldsymbol{X} + \boldsymbol{D}_{\lambda}^{*-1}\right)$, and $\boldsymbol{D}^*$ is a diagonal matrix with diagonal elements $\left(\lambda_j^{-2}+\tau_2^2\right)^{-1}, j = 1, \ldots, p$. It is easy to see that, in their functional form, the conditional distribution of the elastic net parameters is very similar to the lasso, with only minor exceptions.

As far as the shrinkage parameters are concerned, assuming two gamma priors $G\left(r_h, \delta_h\right)$ for $h = 1, 2$, the full conditional distribution of the shrinkage parameters is as follows:

$$\tau_1^2|\boldsymbol{\beta}, \sigma^2, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim G\left(p + r_1, \frac{1}{2}\sum_{j=1}^{p}\lambda_j^2 + \delta_1\right), \tag{3.27}$$

$$\tau_2^2|\boldsymbol{\beta}, \sigma^2, \lambda_1^2, \ldots, \lambda_p^2, \boldsymbol{X}, \boldsymbol{y} \sim G\left(p/2 + r_2, \frac{1}{2\sigma^2}\sum_{j=1}^{p}\beta_j^2 + \delta_2\right).$$

3.9.4. *Horseshoe and the group horseshoe*

Unlike the lasso, ridge and elastic net the horseshoe prior exhibits a pole at zero and polynomial tails, important properties that guarantee good performance in the big data domain (see, for example, Polson and Scott 2010). Assuming the prior structure in equations [3.13]–[3.16], the Gibbs

sampler to estimate the model parameters is rather straightforward. Precisely, Makalic and Schmidt (2015) show that the conditional posterior distribution of the regression coefficients $\boldsymbol{\beta}$ is

$$\boldsymbol{\beta}|\cdot \sim N\left(\boldsymbol{C}^{-1}\boldsymbol{X}'\boldsymbol{y}, \sigma^2\boldsymbol{C}^{-1}\right), \quad \boldsymbol{C} = \left(\boldsymbol{X}'\boldsymbol{X} + \Lambda_*^{-1}\right), \quad \Lambda_* = \tau^2\Lambda$$

where $\Lambda = \text{diag}\left(\lambda_1^2, \ldots, \lambda_p^2\right)$. The conditional posterior distribution of σ^2 is an inverse Gamma given by

$$\sigma^2|\cdot \sim IG\left(\frac{n+p}{2}, \frac{(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})}{2} + \frac{\boldsymbol{\beta}'\Lambda_*^{-1}\boldsymbol{\beta}}{2}\right).$$

The conditional posterior distributions for the local and global shrinkage parameters are also of inverse Gamma type,

$$\lambda_j^2|\cdot \sim IG\left(1, \frac{1}{\nu_j} + \frac{\beta_j^2}{2\tau^2\sigma^2}\right), \qquad j = 1, \ldots, p \tag{3.28}$$

$$\tau^2|\cdot \sim IG\left(\frac{p+1}{2}, \frac{1}{\xi} + \frac{1}{2\sigma^2}\sum_{j=1}^{p}\frac{\beta_j^2}{\lambda_j^2}\right).$$

Finally, the posterior distribution of the auxiliary parameters $\nu_1, \ldots, \nu_p, \xi$ are defined as:

$$\nu_j|\cdot \sim IG\left(1, 1 + \frac{1}{\lambda_j^2}\right), \quad j = 1, \ldots, p \quad \text{and} \quad \xi|\cdot \sim IG\left(1, 1 + \frac{1}{\tau^2}\right). \tag{3.29}$$

Similar to the group-lasso, the extension of the algorithm to the group-horseshoe is somewhat trivial and essentially involves the clustering of the local shrinkage parameters at the group level (see Xu *et al*. 2016).

3.10. References

Adrian, T. and Franzoni, F. (2009). Learning about beta: Time-varying factor loadings, expected returns, and the conditional capm. *Journal of Empirical Finance*, 16, 537–556.

Andrews, D.F. and Mallows, C.L. (1974). Scale mixtures of normal distributions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(1), 99–102.

Arnott, R., Harvey, C.R., Kalesnik, V. and Linnainmaa, J. (2019). Alice's adventures in factorland: Three blunders that plague factor investing. *The Journal of Portfolio Management*, 45(4), 18–36.

Asness, C. and Frazzini, A. (2013). The devil in hml's details. *Journal of Portfolio Management*, 39(4), 49–68.

Asness, C., Friedman, J., Krail, R.J. and Liew, J.M. (2000). Style timing: Value vs. growth. *Journal of Portfolio Management*, 26(1), 50–60.

Asness, C.S., Moskowitz, T.J. and Pedersen, L.H. (2013). Value and momentum everywhere. *Journal of Finance*, 68(3), 929–985.

Avramov, D. (2004). Stock return predictability and asset pricing models. *Review of Financial Studies*, 17(3), 699–738.

Baba-Yara, F., Boons, M. and Tamoni, A. (2018). Value return predictability across asset classes and commonalities in risk premia, Working Paper.

Barberis, N. (2000). Investing for the long run when returns are predictable. *The Journal of Finance*, 55(1), 225–264.

Bianchi, D., Buchner, M. and Tamoni, A. (2019a). Bond risk premia with machine learning. USC-INET Research Paper 19–11, USC-INET Research Paper.

Bianchi, D., Büchner, M. and Tamoni, A. (2019b). What matters when? Time-varying sparsity in expected returns. Working Paper.

Bianchi, D. and McAlinn, K. (2018). Large-scale dynamic predictive regressions. Working Paper.

Binsbergen, V., Jules, H. and Koijen, R.S. (2010). Predictive regressions: A present-value approach. *The Journal of Finance*, 65(4), 1439–1471.

Borri, N. and Verdelhan, A. (2010). Sovereign risk premia, 2010 Meeting Papers 1122, *Society for Economic Dynamics*.

Campbell, J.Y. and Shiller, R.J. (1988). The dividend-price ratio and expectations of future dividends and discount factors. *Review of Financial Studies*, 1(3), 195–228.

Campbell, J.Y. and Thompson, S.B. (2007). Predicting excess stock returns out of sample: Can anything beat the historical average? *The Review of Financial Studies*, 21(4), 1509–1531.

Carvalho, C.M., Polson, N.G. and Scott, J.G. (2009). Handling sparsity via the horseshoe. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, PMCR 5, 73–80.

Carvalho, C.M., Polson, N.G. and Scott, J.G. (2010). The horseshoe estimator for sparse signals. *Biometrika*, 97(2), 465–480.

Chen, L., Pelger, M. and Zhu, J. (2019). Deep learning in asset pricing. Working Paper.

Cohen, R.B., Polk, C. and Vuolteenaho, T. (2003). The value spread. *Journal of Finance*, 78(2), 609–642.

Dangl, T. and Halling, M. (2012). Predictive regressions with time-varying coefficients. *Journal of Financial Economics*, 106(1), 157–181.

Fama, E.F. and French, K.R. (1992). The cross-section of expected stock returns. *Journal of Finance*, 47(2), 427–465.

Fan, J. and Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456), 1348–1360.

Feng, G., Giglio, S. and Xiu, D. (2019). Taming the factor zoo: A test of new factors. *Journal of Finance*.

Feng, G., He, J. and Polson, N. (2018). Deep Learning for Predicting Asset Returns. Working Paper, The University of Chicago.

Feng, G., Polson, N. and Xu, J. (2019). Deep learning in characteristics-sorted factor models, Research Paper 23527, The University of Chicago.

Freyberger, J., Neuhierl, A. and Weber, M. (2017). Dissecting characteristics nonparametrically. CESifo Working Paper Series 6391, CESifo Group Munich. Available at: https://ideas.repec.org/p/ces/ceswps/_6391.html.

Friedman, J., Hastie, T. and Tibshirani, R. (2001). *The Elements of Statistical Learning*. Springer, New York.

Froot, K.A. and Ramadorai, T. (2005). Currency returns, intrinsic value, and institutional-investor flows. *The Journal of Finance*, 60(3), 1535–1566.

Gallant, A.R., Hansen, L.P. and Tauchen, G. (1990). Using conditional moments of asset payoffs to infer the volatility of intertemporal marginal rates of substitution. *Journal of Econometrics*, 45(1-2), 141–179.

Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3), 515–534.

Giannone, D., Lenza, M. and Primiceri, G. (2018). Economic predictions with big data: The illusion of sparsity. Working Paper.

Giglio, S. and Xiu, D. (2017). Inference on risk premia in the presence of omitted factors. NBER Working Papers 23527, National Bureau of Economic Research. Available at: https://ideas.repec.org/p/nbr/nberwo/23527.html.

Gu, S., Kelly, B.T. and Xiu, D. (2018). Empirical asset pricing via machine learning, Research Paper 18-04, The University of Chicago.

Gurkaynak, R., Sack, B. and Wright, J. (2007). The U.S. treasury yield curve: 1961 to the present. *Journal of Monetary Economics*, 54(8), 2291–2304. Available at: https://EconPapers.repec.org/RePEc:eee:moneco:v:54:y:2007:i:8:p:2291-2304.

Hans, C. (2009). Bayesian lasso regression. *Biometrika*, 96(4), 835–845.

Heaton, J.B., Polson, N.G. and Witte, J.H. (2017). Deep learning for finance: Deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1), 3–12.

Hoerl, A.E. and Kennard, R.W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1), 55–67.

Hsiang, T. (1975). A Bayesian view on ridge regression. *Journal of the Royal Statistical Society: Series D*, 24(4), 267–268.

Huang, J.-Z. and Shi, Z. (2018). Determinants of bond risk premia: A resolution of the spanning controversy. Working Papers 2015-12, Penn State University.

Hutchinson, J.M., Lo, A. and Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *Journal of Finance*, 49(3), 851–889.

Jostova, G. and Philipov, A. (2005). Bayesian analysis of stochastic betas. *Journal of Financial and Quantitative Analysis*, 40, 747–778.

Kelly, B., Pruitt, S. and Su, Y. (2018). Characteristics are covariances: A unified model of risk and return, NBER Working Papers 24540, National Bureau of Economic Research. Available at: https://ideas.repec.org/p/nbr/nberwo/24540.html.

Koijen, R.S. and Van Nieuwerburgh, S. (2011). Predictability of returns and cash flows. *Annual Review of Financial Economics*, 3(1), 467–491.

Kozak, S., Nagel, S. and Santosh, S. (2019). Shrinking the cross-section. *Journal of Financial Economics*. Available at: http://www.sciencedirect.com/science/article/pii/S0304405X19301655.

Kuan, C.-M. and White, H. (1994). Artificial neural networks: An econometric perspective. *Econometric Reviews*, 13(1), 1–91.

Kyung, M., Gill, J., Ghosh, M., Casella, G. (2010). Penalized regression, standard errors, and Bayesian lassos. *Bayesian Analysis*, 5(2), 369–411.

Lewellen, J. (2004). Predicting returns with financial ratios. *Journal of Financial Economics*, 74(2), 209–235.

Li, Q., Lin, N. (2010). The Bayesian elastic net. *Bayesian Analysis*, 5(1), 151–170.

Lo, A. (1994). Neural networks and other nonparametric techniques in economics and finance. *AIMR Conference Proceedings*, CFA Institute, 9.

Lustig, H., Roussanov, N. and Verdelhan, A. (2014). Countercyclical currency risk premia. *Journal of Financial Economics*, 111(3), 527–553. Available at: http://www.sciencedirect.com/science/article/pii/S0304405X1300305X.

Makalic, E. and Schmidt, D.F. (2015). A simple sampler for the horseshoe estimator. *IEEE Signal Processing Letters*, 23(1), 179–182.

Marquering, W. and Verbeek, M. (2004). The economic value of predicting stock index returns and volatility. *Journal of Financial and Quantitative Analysis*, 39(2), 407–429.

McCracken, M.W. and Ng, S. (2016). FRED-MD: A monthly database for macroeconomic research. *Journal of Business & Economic Statistics*, 34(4), 574–589.

Menkhoff, L., Sarno, L., Schmeling, M. and Schrimpf, A. (2016). Currency value. *Review of Financial Studies*, 30(2), 416–441.

Messmer, M. (2018). Machine learning and the cross-section of expected stock returns, PhD thesis, University of St. Gallen.

Nalbantov, G., Bauer, R. and Sprinkhuizen-Kuyper, I. (2006). Equity style timing using support vector regressions. *Applied Financial Economics*, 16(15), 1095–1111.

Nozawa, Y. (2017). What drives the cross-section of credit spreads?: A variance decomposition approach. *The Journal of Finance*, 72(5), 2045–2072. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.12524.

Park, T. and Casella, G. (2008). The Bayesian lasso. *Journal of the American Statistical Association*, 103(482), 681–686.

Pastor, L. and Stambaugh, R.F. (2009). Predictive systems: Living with imperfect predictors. *The Journal of Finance*, 4, 1583–1628.

Pesaran, M.H. and Timmermann, A. (2002). Market timing and return prediction under model instability. *Journal of Empirical Finance*, 9(5), 495–510.

Pettenuzzo, D., Timmermann, A. and Valkanov, R. (2014). Forecasting stock returns under economic constraints. *Journal of Financial Economics*, 114(3), 517–553.

Polson, N.G. and Scott, J.G. (2010). Shrink globally, act locally: Sparse Bayesian regularization and prediction. *Bayesian Statistics*, 9, 501–538.

Polson, N.G., Scott, J.G. (2012), On the half-Cauchy prior for a global scale parameter. *Bayesian Analysis*, 7(4), 887–902.

Rapach, D.E., Strauss, J.K. and Zhou, G. (2009). Out-of-sample equity premium prediction: Combination forecasts and links to the real economy. *The Review of Financial Studies*, 23(2), 821–862.

Rapach, D.E., Strauss, J.K. and Zhou, G. (2013). International stock return predictability: What is the role of the United States? *Journal of Finance*, 68(4), 1633–1662. Available at: https://EconPapers.repec.org/RePEc:bla:jfinan:v:68:y:2013:i:4:p:1633-1662.

Rapach, D. and Zhou, G. (2013). Forecasting stock returns. In *Handbook of Economic Forecasting, vol. 2*, Elliott, G. Granger, C. and Timmermann, A. (eds). 328–383, Elsevier.

Rossi, A. (2018). Predicting stock market returns with machine learning, Technical report. Working paper.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Vuolteenaho, T. (2002). What drives firm-level stock returns? *Journal of Finance*, 57(1), 233–264.

Wachter, J.A. and Warusawitharana, M. (2009). Predictable returns and asset allocation: Should a skeptical investor time the market? *Journal of Econometrics*, 148(2), 162–178.

Welch, I. and Goyal, A. (2007). A comprehensive look at the empirical Performance of Equity Premium Prediction. *The Review of Financial Studies*, 21(4), 1455–1508.

Xu, Z., Schmidt, D.F., Makalic, E., Qian, G. and Hopper, J.L. (2016). Bayesian grouped horseshoe regression with application to additive models. In *Proceedings of the 29th Australasian Joint Conference on Artificial Intelligence*, 229–240.

Yao, J., Li, Y. and Tan, C.L. (2000). Option price forecasting using neural networks. *Omega*, 28(4), 455–466.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1), 49–67.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476), 1418–1429.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2), 301–320.

4

The Artificial Intelligence Approach to Picking Stocks

A recent strand of literature has found that machine learning techniques have strong potential for generating a multifactor signal from a large set of (potentially correlated) stock characteristics. The main argument in this literature is that such techniques are suited to the problem, as they are designed to work well in prediction, to efficiently summarize the information contained in a vast set of predictors, identify complex nonlinear relations and avoid overfitting. Models based on neural networks, regression trees and generalized linear regression are shown to outperform some simple traditional benchmarks.

In our work, we start by simulating a nonlinear data generating process for stock returns and illustrating the challenges of calibrating complex nonlinear models. We then fit the models to actual data on global equities over the last 23 years and conduct an extensive out-of-sample analysis to gauge the predictive ability of the proposed methods.

Our chapter complements the existing results in several ways: 1) we extend the analysis to global equities (as opposed to the US market), 2) we focus on an investable universe, 3) we use a proprietary multifactor alpha model as a benchmark, and 4) we compare the resulting strategies based on widely-used performance metrics. The chapter also addresses the common concern that complex non-parametric models are black boxes, i.e. the results are difficult to interpret and therefore auditing decisions based on such models is hard.

4.1. Introduction

The goal of this chapter is to explore the application of machine learning methods to build a multifactor alpha model, a topic which has recently received considerable attention in academia. Rather than providing a general

Chapter written by Riccardo BORGHI and Giuliano DE ROSSI.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

overview of available methods, we focus on a specific application that is of significant interest to many of our clients. In particular, we consider the problem of selecting from a large set of stock-level signals and combining the scores into a single alpha to build quantitative investment strategies.

A few recent academic papers have proposed alternative methodologies to tackle this problem and tested them, as we explain in more detail in our literature review section, with encouraging results. Our contribution can be summarized as follows:

– we apply the proposed methodologies to an investable universe (as opposed to a very broad set of stocks) and test the robustness of the results. The focus is on the profitability of the resulting strategies, for example as measured by the information coefficient (IC), rather than on their predictive ability *per se*;

– we extend the analysis, so far mostly concentrated on US equities, to European stocks as an external validation test;

– we adopt our proprietary Macquarie Quant Alpha model, which has been updated out-of-sample for more than 10 years, as a benchmark. In particular, we investigate whether the machine learning approach departs significantly from the stock selection results of more traditional methods or if, ultimately, both tend to pick similar stocks and generate a similar pattern of returns;

– we feed the resulting alphas to a portfolio optimizer[1] and assess the performance of the model under realistic constraints on style, country, currency exposure and turnover.

Machine learning techniques have recently enjoyed unprecedented levels of popularity in finance. The number of papers that have proposed financial machine learning applications presented at conferences and circulated on public repositories keeps growing, prompting some to doubt whether the phenomenon should be considered a hype. We have come across enthusiastic proponents of the new techniques, who suggest that certain methods are something of a silver bullet that produces better results in most applications. Traditional methods, and particularly linear regression, are sometimes dismissed as obsolete in the era of machine learning.

1 We use the Axioma Portfolio software and its fundamental risk models.

The approach we follow in this chapter is based on the view that machine learning techniques, like other available statistical methods, are tools that are suitable for certain situations and to achieve certain research goals. Does the task of building a multifactor alpha model fit the profile of problems that may be successfully tackled via machine learning techniques? We believe that the answer should be yes for a number of reasons that have been spelled out in the existing academic literature, chiefly Gu *et al*. (2018).

First, machine learning techniques are designed for forecasting purposes. Arguably, being able to forecast stock returns (or to classify stocks as potential winners or losers) is the main goal in building a multifactor alpha model.

Second, machine learning tools are designed to cope with a large number of (potentially highly correlated) predictors. This is the situation typically faced by factor investors as equity "anomalies" proliferate given the constant research work done both in academia and in the industry[2]. Academics currently regard more than 300 alternative signals as well-documented "factors". The vast number of parameters to be estimated may cause some of the traditional statistical tools to produce poor forecasts. As we argue in this report, machine learning methods can drastically reduce the variance of the forecasts by exploiting the trade-off between bias and variance.

Furthermore, equity factors may display complex nonlinear relations with returns and complex interactions. For example, the pattern of returns displayed by stock portfolios sorted on the low beta or low volatility factor is asymmetrical, as argued in Frazzini and Pedersen (2014) and Blitz and van Vliet (2007). Similarly, it is widely accepted that the relation between valuation signals and returns is nonlinear because extremely cheap securities may be value traps. Hence, the function that links expected returns to a typical valuation yield should flatten as the input reaches a certain range.

Complex interactions among different signals are also extensively documented in the empirical evidence available from academic studies, at least since the work of Fama and French (1992) on value and size. Another example is the interplay between size and quality in Asness *et al*. (2018), who find that a portfolio of small cap stocks tends to tilt towards low quality, which detracts from its performance.

2 Harvey *et al*. (2016) analyzed the methodological implications of this phenomenon.

Some of our readers will no doubt be skeptical as to whether the typical financial datasets can afford such a high level of complexity. After all, the main difference between finance and the domains of application in which machine learning has been successful (e.g. e-commerce and computer vision) is that financial data are much noisier. For example, researchers have developed very reliable methods to identify photographs that contain cats. Similarly, a user's taste for books or movies can be inferred with very high levels of precision from his or her history of purchases. Most investors would agree that, except over very short horizons, predicting which stocks will outperform with a comparable level of accuracy is arduous, even with extensive information on company fundamentals and past price behavior. Predictive models of stock returns typically rely on a large set of weak predictors to be combined in an optimal way, with information coefficients that rarely exceed 5%.

There are several obvious reasons for this difference in the signal-to-noise ratio. Decades of theoretical work on financial markets have analyzed the implications of market efficiency. As financial markets develop, predictability of stock returns tends to be progressively eliminated. No such mechanism is at play in e-commerce data. Furthermore, financial data is characterized by structural breaks which permanently alter the relation between predictors and stock returns. Finally, financial variables are driven by macro factors which evolve slowly over time. As a result, we typically do not have enough data to train complex models from past business cycle fluctuations.

Hence, the fourth reason for considering a machine learning approach in this context is the availability of a range of techniques designed to mitigate overfitting. Two notable examples are parameter regularization and the idea of splitting the sample into training, validation and testing subsamples. The former limits the flexibility of the model, thereby preventing the optimizer from overfitting input data by choosing "extreme" parameter values. The latter forces the optimizer to use two separate datasets to estimate model parameters and calibrate hyperparameters (e.g. the rate at which the optimizer "learns" from the data).

A fifth reason that has sometimes been put forward in the literature is the need to avoid crowded trades in the quant space. Signals used by quantitative investors are likely to be highly correlated and therefore, the argument goes,

positions are likely to have large overlaps. A new way to process the available data, i.e. the machine learning approach advocated by these papers, may be key to generating portfolios that avoid crowding.

We do acknowledge that each of these points can be addressed with more traditional tools, particularly tools from non-parametric statistics[3]. However, in this chapter, we aim to explore, while retaining a healthy dose of skepticism, the application of machine learning to alpha modeling. In particular, we carry out a large regression analysis with the goal of predicting monthly stock returns for two liquid universes, MSCI USA and MSCI Europe, between 1995 and 2018. The set of potential predictors includes most of the widely-used quantitative factors such as value, price momentum, earnings momentum, quality and low risk. Alongside linear regression, we consider a number of models that have been popularized in the machine learning literature, including regularized regressions (LASSO, ridge, elastic net), neural networks, boosted tree models and random forests. Each model generates a firm-level alpha signal that can be used for stock selection in a trading strategy.

The main empirical results can be summarized as follows. First, machine learning techniques have strong potential and, even when compared to a proprietary model with a successful track record like our Macquarie Quant Alpha model, have enough predictive power to generate attractive risk return ratios. In our out-of-sample analysis, we find that an equally weighted long–short portfolio of stocks sorted by their predicted one-month return would have generated information ratios (IR) of 1.18 for US stocks and 1.32 for European stocks. The period used for the backtest is a challenging one for factor investing, as demonstrated by the weak performance of the standard style factors used in academic research. This notwithstanding, over the same out-of-sample period, an optimized strategy without turnover constraints would have delivered information ratios of 1.26 in the USA and 2.16 in Europe before transaction costs.

Finally, we find (in line with the empirical results reported in academic papers) that price momentum factors and technical indicators feature prominently in the optimal signal combination. Nevertheless, the drag represented by transaction costs does not wipe out the outperformance of long–short strategies based on the proposed methodology.

3 For example, Horowitz *et al*. (2000) used linear spline regressions to analyze nonlinearities in the size premium, while Connor *et al*. (2012) used a semiparametric estimator to allow for nonlinearities in the Fama–French (1992) model.

4.2. Literature review

Machine learning in finance is not a new field. Neural network applications, for example, were popular in the 1990s, so much so that a collection of articles on the topic appeared in 1992 as a book edited by Robert Trippi and Efraim Turban (Trippi and Turban 1992). The specific application we deal with in this chapter also attracted some attention in the 1990s, as demonstrated by early works such as Mezrich (1994).

Among the recent strand of empirical papers that have attempted to create multifactor models of stock returns, Gu *et al*. (2018) is arguably the most comprehensive. The authors use a variety of nonlinear non-parametric techniques (including boosted regression trees, neural networks and random forests) to predict monthly stock returns based on approximately 90 "factors" and eight macro variables. Their main conclusion is that machine learning methods, and neural networks in particular, significantly outperform both a simple three-factor model such as the Fama–French (1992) model and a linear regression model that uses all available factors.

The authors have admitted that the least squares benchmark is a very weak one because the OLS estimator struggles to deal with almost 1,000 highly correlated independent variables. This is one of the key advantages of some of the models considered in this chapter: by introducing a small bias in the forecast, they can improve predictive accuracy when traditional linear estimators have high variance. Selecting a small subset of predictors or grouping and averaging within each group are simple solutions that achieve a similar result, i.e. reducing the variance of the estimator by setting some of the coefficients to zero.

Gu *et al*. (2018) found that the success of their forecasting models is driven predominantly by cross-sectional information, not by the information contained in macro variables. In particular, among the various factors, momentum, liquidity and volatility, as opposed to balance sheet variables, prove to be most important in achieving predictive power. It is worth noting that all these variables are updated at monthly frequency, while company fundamentals are updated once per year (see Green *et al*. 2013a, 2013b). The frequency of updating seems to boost the out-of-sample R^2. One of the main results in Gu *et al*. (2018) is that machine learning methods learn

efficiently from a vast[4] dataset characterized by a large cross-section of firms, a large number of characteristics and a large time series dimension.

It is worth mentioning that they do not exclude penny stocks from the sample, a very unusual choice for an asset pricing paper. The authors do, however, report that the model has a good out-sample fit in the large cap portion of their sample, even though (unsurprisingly) the equally weighted version of their simulated strategies outperform the value-weighted ones.

Krauss *et al*. (2017) also considered a range of different nonlinear models (neural networks, gradient-boosted trees and random forests). Their set of predictors only includes past prices and the prediction horizon is daily. Unlike Gu *et al*. (2018), Krauss *et al*. (2017) set up the problem as a classification one, i.e. identifying outperforming stocks. Their main conclusion is that a statistical arbitrage strategy based on their daily forecasts would have generated significant excess returns on a passive benchmark, even though the profitability of the strategy appears to decline over time. Their analysis of variable importance suggests that the model loads predominantly on short-term features, i.e. lagged returns from day $t-6$ to day $t-1$.

Coqueret and Guida (2018) used a tree-based approach to predict which stocks will outperform using a set of 30 characteristics. Guida and Coqueret (2019) applied extreme gradient-boosted trees to build enhanced diversified equity portfolios. Their goal was to predict the 12-month (sector-relative) performance of individual stocks using a large number of features. They reported significant gains in risk-adjusted performance on a simple multifactor portfolio. The factor exposure of the resulting strategy, nevertheless, remains consistent with a traditional multi-factor portfolio. The authors concluded that tree-based models can be useful to avoid crowding in quantitative strategies.

Guida and Coqueret (2018) used a range of machine learning methods, including neural networks, random forests and extreme gradient-boosted trees to predict 12-month stock returns.

Another recent paper has focused on specific families of machine learning models. Moritz and Zimmermann (2016) proposed the use of

4 At least by the standards of financial economics.

"tree-based conditional portfolio sorts". In practice, their approach consists of training a random forest model on 86 fundamental firm characteristics and a set of lagged monthly stock returns. Again, the model displays strong predictive power on monthly returns and lagged price information appears to be the main driver of the result.

Several recent papers (DeMiguel *et al.* 2018; Freyberger *et al.* 2017; Feng *et al.* 2019; Messmer and Audrino, 2017) have adopted the LASSO or one of its generalizations as a tool to select stock-level predictors from a large set of candidate variables.

Another strand of literature has focused on artificial neural networks. Feng *et al.* (2018) proposed the use of deep neural networks to build composites of 57 characteristics. Abe and Nakayama (2018) adopted a similar approach to predict monthly returns to Japanese stocks using 25 stock characteristics. Messmer (2017) is a further example, using 68 firm-level variables in an application to US data.

While all the neural network papers mentioned above have treated the forecasting problem as a cross-sectional one, Fischer and Krauss (2018) focused on time series properties. In particular, they applied long–short term memory networks to predict daily stock returns of S&P 500 constituents and reported encouraging results.

Some empirical results are common to most of the papers we have reviewed. The main conclusion is invariably that machine learning techniques have strong potential, as demonstrated by the evidence that they outperform some simple traditional benchmarks. The most complex models (e.g. deep neural networks) are typically outperformed by simpler ones, as long as sufficient flexibility is allowed. Price momentum factors feature prominently in the optimal signal combination. The number of useful predictors and relative weight change significantly over time.

We conclude this section by mentioning three themes that emerge from the papers we have reviewed. The first theme is the choice between a regression and a classification framework. The returns prediction problem is approached in two alternative ways:

– regression, to rank stocks according to their forecasted returns, as in Gu *et al*. (2018);

– classification, to rank stocks according to their probability of outperforming their universe or their sector, as in Guida and Coqueret (2019), Krauss *et al*. (2017) and many others.

The second important theme concerns style timing. Should the predictive model learn how style returns are affected by macroeconomic conditions? More broadly, should the model adapt to changing conditions in the market? One of the characteristics of financial data that make prediction a challenging task is model instability.

Gu *et al.* (2018), as we explain in more detail below, choose to augment their set of cross-sectional characteristics with a number of interaction terms that involve a firm-level variable and a macro variable. Hence, their model is sufficiently flexible to incorporate style timing. It is trained on a sample that spans a long period of time (at least 30 years), in order to capture the dynamics efficiently. Other papers either ignore the issue altogether (essentially treating the problem as a cross-sectional one) or use much shorter in-sample training periods to allow the model parameters to adapt dynamically.

Finally, we highlight a theme related to model calibration. k-fold cross-validation is widely used for cross-sectional models in which the sample can be naturally split into a small number of subsamples. In the case of stock returns, however, the methodology needs to account for the complex dynamics of the data generating process. As we explain in more detail in the section on model calibration, several papers have dispensed with cross-validation and used a simpler "validation" approach. For example, Gu *et al.* (2018) split the in-sample observations into two subsamples along the time dimension. The earlier subsample is used as the training set, while the later ones constitute the validation set. Abe and Nakayama (2018) adopted a similar technique.

4.3. Data

4.3.1. *Equity factors*

Our empirical analysis uses data for all constituents of MSCI USA and MSCI Europe from January 1995 to November 2018. Models are trained in each universe separately. Our goal is to predict monthly total stock returns in a large regression exercise.

The so-called features matrix (i.e. the independent variables used for prediction) consists of roughly 200 stock-specific signals from our factor library. For details on the methodology behind factor scores, normalization and update frequency, see the Macquarie Quantitative Research Report (2008, 2010). From the full dataset, we whittled down the number of features by applying the data cleaning filters described below.

The surviving factors represent a broad selection of the risk premia or anomalies that have been documented in the academic literature:

– value factors, which we further classify as relative value (e.g. the internal rate of return – IRR), defensive value (e.g. dividend yield) and cyclical value (e.g. the book-to-price ratio);

– quality factors, for example profitability ratios such as ROE (return on equity);

– price momentum factors, for example lagged 6- and 12-month returns;

– earnings momentum factors;

– technical factors, for example Williams %R;

– low risk factors such as (low) price volatility and (low) market beta;

– liquidity factors;

– factors related to capital structure such as the debt-to-equity ratio.

For factors that depend on analyst forecasts, whenever consensus data is available, we include versions of the factor based on different forecasting horizons and trailing data. For example, the book yield signals included use FY0, FY1, Last Twelve Months and Next Twelve Months forecasts. The scores are typically highly correlated – a challenge for any multifactor model is to process the information efficiently in this framework.

We also trained the models with an extended features matrix that contains both stock-specific variables and their interaction with eight macro variables from Goyal and Welch (2008)[5]. We included them in the model to primarily contemplate the nonlinear interaction between stock-specific features and indicators of the state of the economy; however, we found that the macro variables do not add any forecasting power. This might be due to the small

5 We use eight macroeconomic predictors at the market-aggregated level, following the variable definitions detailed in Goyal and Welch (2008).

time series dimension in our empirical application, which is not sufficient to capture enough business cycles for the model to learn how to rotate styles. However, Gu *et al*. (2018), using 18 and 12 years of, respectively, training and validation, also found that macro variables are not relevant. Furthermore, Goyal and Welch (2008) showed that in the last 30 years, they have performed very poorly. Thus, we use only stock-specific signals as input features.

4.3.2. *Data cleaning*

Every month, we retain the signals (features) with at least 60% universe coverage. For the remaining features, following Gu *et al*. (2018), missing data are replaced with their cross-sectional median (from a 1%-trimmed distribution to reduce the impact of outliers). This procedure adds a penalty to missing data, i.e. according to that specific signal the stock will be placed at the center of the distribution[6]. Finally, each feature is cross-sectionally normalized to be between -1 and 1 using a formula based on ranks. This procedure has two advantages: first, it eliminates the impact of outliers without the need for winsorizing/trimming the distributions. Second, it prepares the data for neural networks, whose results depend on the scale of the inputs. We perform the same transformation to the response variable. Thus, this ensures that the cross-sectional standard deviation of the features and response are homogeneous and constant over time. We also trained the models with the raw response variable on a small sample of data, but we found a massive improvement in the smoothness of the error function when using transformed returns; that is, with untransformed response the models struggle to learn and the MSE (mean squared error) in validation immediately explodes, leading to the model predicting all returns with a constant.

4.3.3. *Features used for training and prediction*

After the cleaning procedure, we are left with a time-varying number of signals. For instance, for the MSCI USA index, the number of signals goes

6 Another option is to set NAs equal to zero after z-scoring the data, but this could place the stock in the lower part of the distribution if this has a strong positive skewness. However, this would work well if a normal distribution is forced on the data with an iterative procedure. Further details on the impact of data cleaning procedure on model performance can be found in the Macquarie Quantitative Research Report (2010).

from 128 at the beginning of the sample to 197 at the end of the sample. With more consensus data available, both the number and the quality of the signals improve with time.

However, to avoid look-ahead biases, we use an expanding window estimation. Hence, following Gu *et al*. (2018), we only use the features that were available since the very beginning of the sample (January 1995). It is worth noting that this constraint can be relaxed and also that linear models can be adapted to deal with missing data. As a result, we view our results on the predictive ability of the machine learning as conservative.

4.4. Model specification and calibration

4.4.1. *Models*

This section provides an overview of the models used in our empirical analysis. Our choice of machine learning techniques draws from the available empirical literature, particularly Gu *et al*. (2018). All models have been calibrated using the open-source machine learning library H2O, through the R package "h2o", which is available from CRAN. We start with a general statement of the forecasting problem.

We express the estimation problem as a regression design using an overarching model that encompasses all the linear and nonlinear specifications that we will estimate. The overarching model can be written as:

$$r_{i,t+1} = E_t\left(r_{i,t+1}\right) + \varepsilon_{i,t+1}$$

where $E_t\left(r_{i,t+1}\right)$ is a function that depends neither on i nor on t. The easiest example of this function is a panel model with homogeneous intercept and slopes, which can be written as:

$$r_{i,t+1} = \beta_0 + \sum_{j=1}^{P} \beta_j X_{j,i,t} + \varepsilon_{i,t+1}$$

for $i = 1, \ldots, N$ and $t = 1, \ldots, T$. P is the number of predictors (also called "features"). This model can be estimated with standard OLS, even with P larger than N and T, as long as $NT > P$. Gu *et al*. (2018) used a similar

panel model, with additional industry-specific dummies[7]. The objective functions that we minimize are all based on pooled residuals, calculated either in training or validation.

4.4.1.1. *Benchmark models*

Macquarie Alpha model: the Macquarie Alpha model is Macquarie's global stock selection model, which has been successful across different markets and has generated significant outperformance since its launch in 2008. Our Alpha model uses a library of almost 200 quantitative factors and combines them to obtain a single stock picking score that ranks all stocks relative to each other within a given universe. Factor weights are dynamically selected in order to maximize risk-adjusted performance over the three-month horizon while controlling for turnover and maintaining factor diversification.

Forward-stepwise selection: this method, initially suggested by Efroymson (1960), can be considered as a naive machine learning approach. It uses a "subset selection" procedure called forward-stepwise regression to choose the next best predictor of future stock returns among a set of variables (features), starting from a regression with an intercept. It uses the in-sample Bayes information criterion (BIC) and we adapted this as an early stopping criterion with patience of 5. Because of the forward nature of the algorithm, we can stop the search when the BIC does not improve for five additional predictors. Hastie *et al*. (2009) showed that in-sample information criteria are good estimates of the expected forecast error and they are good statistics to use when not enough data is available for splitting the sample between training and validation.

4.4.1.2. *Penalized linear models (LASSO, ridge and elastic net)*

Let us ignore the time series dimension to simplify notation. The elastic net augments the least squares loss function with L_1 and L_2 penalties. It encompasses the LASSO and ridge as special cases. To find the elastic net solution, we solve:

$$min_w \frac{1}{2N}\sum_{i=1}^{N}(r_i - w'X_i)^2 + \lambda\left(\alpha\sum_{j=1}^{P}|w_j| + \frac{1}{2}(1-\alpha)w'w\right)$$

7 We also estimated a version of the models with sector dummies, but since the results are very similar, we chose to use the most parsimonious model.

where λ and α are hyperparameters to be calibrated. λ is the strength of the regularization and α is the elastic between the two penalties. If we set $\alpha = 1$, we find the LASSO solution, while for $\alpha = 0$ we have the ridge estimator. To be consistent, we solve all three models using the "cyclical coordinate descent" method, even though the ridge estimator has a closed-form solution. Friedman *et al*. (2010) showed that coordinate descent is fast and efficient for models with elastic net penalties, especially when the solution is searched along the regularization path, i.e. find the optimal λ for a fixed value of α.

For all three models, λ is calibrated using early stopping and exploring up to 100 values starting from λ^*, which is defined as the smallest value of λ that is large enough to set all slopes to zero, i.e. only an intercept is left in the model. As λ is decreased, the regularization is relaxed. The algorithm stops either when 100 values of λ are explored or if the early stopping parameter is satisfied.

Thus, the LASSO and ridge problems are solved by finding the optimal λ with $\alpha = 1$ and $\alpha = 0$, respectively. For the elastic net, we use a Cartesian search to find α, searching in the set $\{0,\ 0.01, \ldots, 1\}$.

4.4.1.3. *Generalized linear model (GLM)*

The traditional GLM adds nonlinearities to a linear model, adding spline series expansions of the predictors. We implement a different version of GLM, where we add to the model the cross-product of the features:

$$r_{i,t+1} = \beta_0 + \sum_{j=1}^{P} \beta_j X_{j,i,t} + \sum_{\substack{j,r \\ r>j}}^{P} \gamma_{j,r} X_{j,i,t} X_{r,i,t} + \varepsilon_{i,t+1}$$

We then add a LASSO penalty to the objective function to regularize the model.

4.4.1.4. *Regression trees*

Regression trees are simple methods to split the response according to the value of the features. The tree grows up to a maximum depth that has to be specified, representing one hyperparameter to calibrate. In our framework, this boils down to dividing the stocks into categories according to the value of the features and finally taking the average returns of the stocks remaining in the leaves, i.e. the categories in the final branch of the tree. Thus, if two stocks end up in the same category, their predicted return for that specific

tree will be equal. Combining several trees, however, makes the cross-sectional predictions smoother.

A tree with J leaves and for a given depth can be expressed as:

$$T(X_i;\Theta,J)=\sum_{j=1}^{J}\theta_j I\left(X_i\in R_j\right)$$

where θ_j is the prediction for region R_j and Θ is the parameter set. The tree is built as follows: for each branch (from 1 to the maximum depth L) and for each feature, we build a histogram. At each possible split, the stocks are partitioned according to the thresholds of the histogram and for each split we calculate the loss (also called "impurity"). The optimal split (which feature and at which threshold of the histogram) is calculated as the one that minimized the loss. Similar to Gu *et al.* (2018), we use the L_2 impurity for each branch of the tree, which implies that the optimal choice of θ_j is simply:

$$\widehat{\theta_j}=\frac{1}{|R_j|}\sum\nolimits_{X_i\in R_j} r_i$$

where $|R_j|$ is the number of stocks in partition R_j. In the next step, the same algorithm is run on each branch using the remaining features. The algorithm stops when there is no more data to split. We set this to a minimum of 10 observations for a leaf in order to split (H2O default is 10). Thus, a single tree will stop splitting either if there is no more data to split, if it reaches its maximum depth or if there are no splits that satisfy a "minimum split improvement" parameter (H2O default is 10^{-5}).

The type of histogram that is used to split has an influence on the algorithm. For all tree-based models, we grow each tree of the ensemble using a different type of histogram ("uniform adaptive", "quantiles global" and "random"), so that we can diversify the "histogram risk", i.e. the risk of using a histogram that is not appropriate for the feature distribution.

4.4.1.5. *Gradient-boosted trees (GBM) and random forest (RF)*

Without regularization, regression trees are prone to overfit. One major problem is the large variance of their prediction. A small change in the data can often result in a very different series of splits. The major reason for this

instability is the hierarchical nature of the process: the effect of an error in the top split is propagated down to all of the splits below it.

In order to mitigate this problem, we implement two types of regularization: boosting and bagging. Both methods consist of an "ensemble of trees". In H2O, the gradient-boosted tree is called a gradient-boosted machine (GBM). We will use this same term throughout the chapter.

The first regularization method for regression trees is called *boosting*, which results in a technique called gradient-boosted regression tree, or gradient-boosted machine. Friedman *et al*. (2000) extended boosting from classification problems to the gradient-boosted regression tree. We first grow a shallow tree (e.g. depth $L = 1$). This tree is a weak learner. Then, we grow a second tree of the same depth L using the residuals from the first tree. The returns prediction is calculated as the sum of the predictions from these two trees, with the residual-based prediction shrunken by a factor between 0 and 1 to avoid overfitting the tree to the residuals (i.e. similar to a weighted average of predictions from return tree and residual tree). The algorithm is iterated up to a total of B trees, which results in an ensemble of trees with more stable predictions. For a fixed tree depth, this can be expressed as:

$$BT_B(X_i; v) = \sum_{b=1}^{B} v\, T(X_i; \Theta_b, J)$$

where the trees are calculated in a forward stagewise manner. At each step, we solve:

$$\widehat{\Theta}_b = argmin_{\Theta_b} \sum_{i=1}^{N} \left(r_i - BT_{b-1}(X_i) - T(X_i; \Theta_b)\right)^2.$$

We use the stochastic version of gradient-boosted machine, where each tree is grown using a sample of 50% of rows of the features matrix. Hastie *et al*. (2009) claimed that this version of gradient-boosted machine outperforms random forest in prediction. Gradient-boosted machine trains very quickly, so we search a grid of possible shrinkage factors using Cartesian search. In addition, we use an early stopping rule to decide on the number of trees and maximum depth. The algorithm first tries to grow and combine up to 500 trees with depth equal to 1, then it tries to grow and combine up to 500 trees with depth equal to 2, etc. We limit the tree depth to $L = 5$ because gradient-boosted machine is based on combining several shallow trees.

The second regularization for regression trees is called *bagging*, which results in a technique called random forest. The tree-building algorithm is the same as the one used in gradient-boosted machine, where each tree is grown using a different type of histogram. The main difference is that gradient-boosted machine combines several shallow trees, while random forest builds uncorrelated trees that are potentially deeper and then averages their predictions. Bagging generates the prediction:

$$\hat{r}_i = \frac{1}{B}\sum_{b=1}^{B} T_b(X_i)$$

i.e. we average the predictions generated by individual fitted trees.

Each tree generated in bagging is identically distributed[8], so the expectation of an average of B trees is the same as the expectation of any one of them. This is in contrast to boosting. Thus, the bias of bagged trees is the same as that of the individual trees and the only hope of improvement is through variance reduction. The variance of B identically distributed (but not necessarily independent) random variables, each with variance σ^2 and positive correlation ρ, is equal to:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2.$$

The idea of random forest is to decorrelate the trees considering only a randomly drawn subset of predictors for splitting at each potential branch, reducing the first term in the formula. Random forest is much slower to train than gradient-boosted machine because the trees can go much deeper than in gradient-boosted machine.

4.4.1.6. *Neural networks (NN)*

Neural networks are nonlinear models that allow for complex interactions among predictors. We use a feed-forward fully connected neural network (also called the sequential neural network) with three hidden layers consisting of 32, 16 and 8 neurons, respectively, which has been found to have good predictive ability in Gu *et al*. (2018). A simplified version of the neural network with a single layer and M neurons can be written as:

$$\hat{r}_i = \beta_0 + \beta' Z$$

8 Trees in the forest are independent because at each level of splits we randomly sample a fraction of the features from the whole pool of P possible predictors.

$$Z_m = f(\alpha_{0m} + \alpha'_m X_i),\ m = 1, \ldots, M$$

Neural networks typically require estimation of a large number of parameters, in the formula above $M(P+1)$ alphas and $M+1$ betas, where P is the number of predictors. As activation function, f, we use the popular rectified linear unit (ReLu), which is defined as:

$$f(x) = ReLU(x) = \max(x, 0)$$

The activation function is used in each of the hidden layers. The last layer is still dense but there is no activation function, i.e. the output from the last hidden layer is combined linearly to form the return prediction.

We estimate the neural network parameters by minimizing the penalized L_2 objective function of prediction errors. Hornik *et al*. (1989) showed that, under reasonable regularity conditions, this approach produces consistent and asymptotically normal forecasts. Solving the optimization is computationally intense, and a common solution is to use stochastic gradient descent (SGD), which evaluates the gradient from a small random subset of the data, instead of using the entire training sample. We use a modification of SGD called ADADELTA, proposed by Zeiler (2012). This algorithm combines the benefits of learning rate annealing and momentum training to avoid slow convergence. This is extremely useful for loss functions with local minima and long plateaus. ADADELTA is efficiently implemented in H2O (*adaptive rate*) and only requires two hyperparameters to be calibrated, compared to seven parameters that need to manually calibrate the adaptive rate.

To avoid overfitting, we use the following regularization methods:

– hidden dropout of 50% and input dropout of 20%;

– L_1-penalty of the weight parameters;

– early stopping;

– learning rate shrinkage;

– ensembles.

In particular, we create an ensemble of five neural networks, each calibrated with a random grid search of the hyperparameter space. The five predictions are combined using either a simple average or linear weights

proportional to the inverse of the MSE in validation. In general, we prefer equally weighted averages to diversify the *ex ante* model risk. More information on the calibration can be found in the next section.

4.4.2. *Model calibration*

In this section, we briefly review the current approaches to train machine learning models, and we present the method we will use in this chapter. Each model requires, in addition to parameter estimation, the calibration of a set of hyperparameters that determine its architecture and "complexity". For instance, a stepwise selection method has one hyperparameter that is the number of predictors, a LASSO algorithm has to calibrate the L_1-penalty, etc. Within the model structure defined by the hyperparameters, we estimate the parameters that govern the model such as weights in the neural network, slopes in the linear models and splitting thresholds in tree models.

To this end, the sample period is typically split into three subsamples referred to as *training*, *validation* and *test* samples. Two main approaches have been adopted in the empirical literature.

The first approach ("validation") consists of splitting the in-sample observations into two subsamples along the time dimension. The earlier subsample is used as the training set, while the later ones constitute the validation set.

Gu *et al*. (2018) adopt the "validation method", using a long time series, expanding window estimation and interactions of stock-specific variables with macro variables. This implies embedding a style rotation feature in the model, but it assumes that the training sample comprises a full business cycle, so that the model can learn how the various types of stocks (e.g. growth and value) have performed during different phases of the business cycle. Gu *et al*. (2018) used a sample of 30,000 stocks listed in US markets from 1957 to 2016, and they divide the sample in 18 years of training sample (1957–1974), 12 years of validation (1974–1986) and 30 years of out-of-sample testing. They refit the model once per year, i.e. they keep the parameters and hyperparameters fixed for one year and create monthly forecasts using the new features that become available. After one year, they refit the model using an expanding window training sample (18, 19… years) but whilst holding the validation period to 12 years at the end of the sample.

To calibrate the hyperparameters, they build various types of objective functions using the pooled residuals calculated in the validation period that is always at the end of the sample. This ensures that we calibrate the model according to its forecasting ability in the most recent part of the sample.

In the second approach ("k-fold cross-validation"), the in-sample information is split into k subsamples ("folds"). In each of the k steps, one of the folds is selected to act as the validation set, while the others make up the training set. This technique poses some notable challenges. First, it requires a large amount of data for training to be effective in the subsamples. Second, it is typically computationally intensive. Third, when the time dimension matters, like in our setup, it may be unsuitable due to potential structural breaks.

Guida and Coqueret (2019) used a modification of the "cross-validation method", estimating the model using a shorter training/validation window, essentially assuming that there is not enough data to split the sample between training and validation in temporal order. This method has the advantage of using more recent feature information, but it is computationally more intensive. Guida and Coqueret (2019) used a three-year rolling window to fit a tree model through XGBoost, where the first two years were used for training and the last one was used for validation. In this case, a k-fold cross-validation is performed by splitting the universe of stocks (both in the training and validation time periods) into k random baskets. For instance, Guida and Coqueret (2019) used a five-fold cross-validation, where the model was fitted on 4/5 of the universe using the training sample and cross-validated in the remaining 1/5 of the stocks using the validation sample. This is different from the traditional cross-validation method used in the machine learning literature that would imply "testing in the past". In general, the validation is done on 20% of the sample, so $k = 5$ is a good rule of thumb.

We broadly follow the structure of Gu *et al*. (2018) and train the models using an expanding training/validation window that starts with a length of 16 years, from January 1995 to December 2010. The validation sample always consists of eight years at the end of the in-sample window. After calibrating the models, we keep all the weights and hyperparameters fixed for one year and use them to generate monthly stock returns prediction based on the new data that becomes available out of sample. Thus, our test sample is 12 months. The training window is then increased by one year and the process is repeated. Figure 4.1 provides a graphical representation of this process.

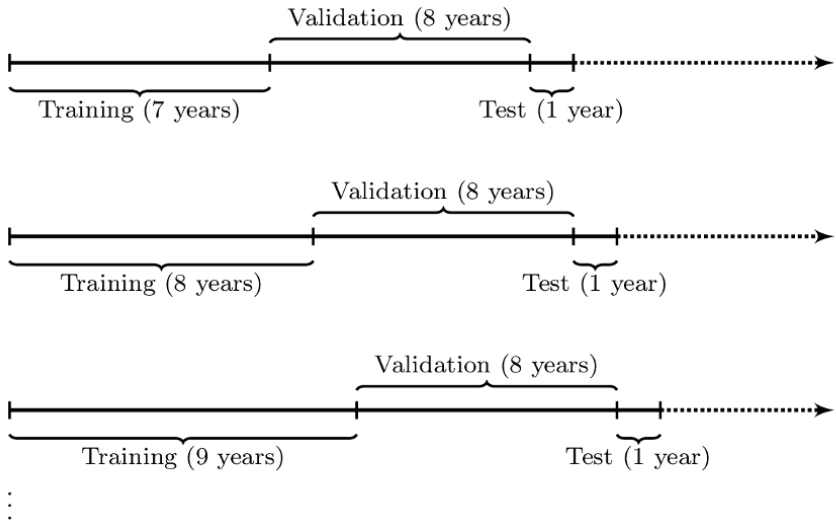


Figure 4.1. *Sample split in training, validation and test (source: Macquarie Quantitative Strategy, January 2019)*

4.5. Predicting US stock returns

In this section, we analyze the stock return predictions for the constituents of MSCI USA from 1995 to 2018. Since both features and response are normalized, we do not predict the raw returns but we will predict their rankings. Thus, we assess the model performance using information coefficients (ICs) instead of the R^2 goodness of fit. We then simulate a simple strategy that consists of sorting stocks based on the models' predictions, buying the stocks in the top quintile and selling those in the bottom quintile. The strategy is equally weighted. We compare the strategies based on the stock-specific prediction from each model plus an "ensemble" model, and average the predictions from gradient-boosted machine, random forest and neural network. For the neural network model, we use both the equally weighted average (NN_ew) of the predictions from five neural networks and the weighted average of five predictions, with linear weights that are proportional to the inverse of the MSE in validation (NN_msew).

4.5.1. *Information coefficients*

Figure 4.2 reports average ICs in validation and testing. The validation period is always an eight-year sample before the test. Each test sample is one year, where we predict one-month forward returns using the models trained with data up to the end of December. The out-of-sample period starts in January 2011 and ends at the end of November 2018.

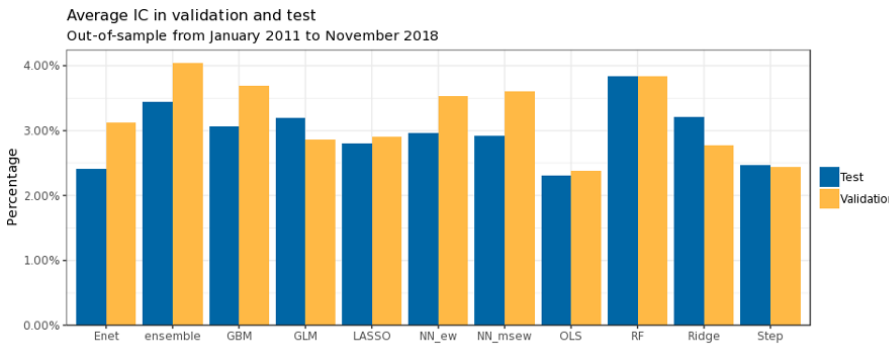


Figure 4.2. *Information coefficients, MSCI USA (source: Macquarie Quantitative Strategy, January 2019)*

OLS is the worst performing model both in validation and test. This is not surprising, and it is probably due to the multicollinearity among features. This is confirmed by the very good performance of the ridge estimator, which is often used in corporate finance applications to deal precisely with correlated regressors. Also, a simple LASSO estimator outperforms OLS, while elastic net shows signs of overfitting, with some poor generalization skills from validation to test. The surprising generalization ability of the ridge model suggests that there is a benefit in including all available variables, instead of selecting a few of them with penalized linear models. A simple GLM that includes all features plus their pairwise interactions, with L_1-penalty, performs as well as the ridge model. The step model (forward stepwise selection) only marginally beats OLS, although it generalizes better.

The best performing model for this universe is the random forest, which displays a large average IC in validation (almost 4%) and an almost identical IC in test, showing very good generalization skills. The gradient-boosted machine performs better than most linear models, but it overfits more than the

ridge model and GLM. The equally weighted neural network and MSE-weighed neural network weighting schemes show very similar results, with the MSE-weighed neural network displaying slightly more overfitting tendency. We suggest to always use equally weighted schemes to diversify the *ex ante* model risk. The performance of neural networks for the MSCI USA universe is very similar to that of gradient-boosted machine. They show some signs of overfitting, but they still reach a decent average 3% IC in test samples.

Note that OLS, even though it is the worst model for this universe, does not perform as poorly as in Gu *et al*. (2018), who reported very negative R^2 for predictive regressions using the OLS model. This is because we only use stock-specific features, while they add interactions with macro variables. It is no surprise that in such a framework, OLS is not able to disentangle the relevant variables and noise.

To visualize the model performance over time, Figure 4.3 plots the average IC in validation and test samples at each re-estimation date. We also plot the IC obtained with OLS as a reference point. Since we use an expanding training window, from these plots we can also visualize the dependence of the model performance on the amount of data available. From the left panel, we can see that the more data, the better: as the training window expands, the IC in the latest eight years becomes higher and higher. The models also perform well in test, with average (yearly) ICs always positive until 2017, i.e. until December 2017. However, this does not *always* translate into a positive average IC out-of-sample, with the average IC turning negative in 2018.

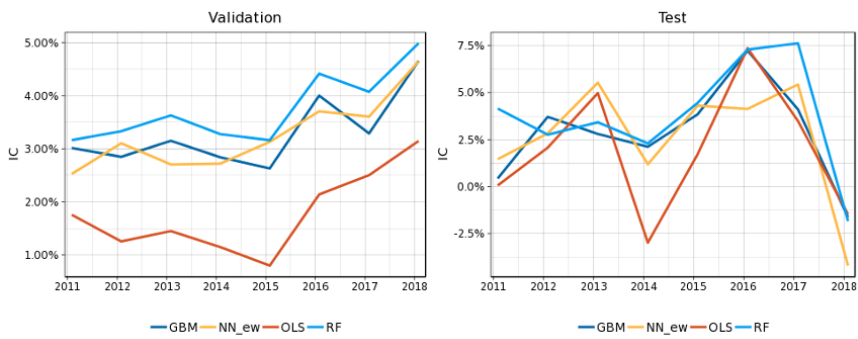


Figure 4.3. *Average IC in validation (eight years) and test (one year), MSCI USA (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.5.2. *Long–short strategy*

Figure 4.4 reports the performance metrics for the equally weighted long–short strategy. The backtest period is relatively short but data availability is limited given the need to train each model on a long initial subsample. We report annualized average returns and volatility, information ratio (IR) and maximum drawdown (from peak to trough). To analyze the distribution of each strategy's return, we compute skewness, excess kurtosis, hit rate (percentage of months where the long–short returns are positive), t-statistics for the hypothesis that average long–short returns are equal to zero and relevant p value assuming a t-distribution with $T - 1$ degrees of freedom.

Model	Mean return	Volatility	IR	Max drawdown	Skewness	Excess Kurtosis	Hit rate	Number of obs	t-Test	p value
RF	8.49%	6.10%	1.39	10.70%	-0.10	0.14	63%	95	3.92	0.00
ensemble	7.33%	6.24%	1.18	10.10%	0.13	0.71	61%	95	3.31	0.00
GBM	6.34%	5.60%	1.13	9.60%	0.13	0.39	59%	95	3.19	0.00
Ridge	6.29%	7.13%	0.88	11.00%	0.09	0.10	57%	95	2.48	0.02
NN_ew	6.04%	7.08%	0.85	10.60%	0.10	1.09	61%	95	2.40	0.02
ALPHA	6.29%	7.63%	0.82	15.00%	-0.68	0.69	67%	95	2.32	0.02
NN_msew	5.58%	6.99%	0.80	10.80%	0.11	0.90	61%	95	2.24	0.03
GLM	5.49%	7.07%	0.78	9.20%	0.13	0.60	58%	95	2.19	0.03
LASSO	5.36%	7.04%	0.76	8.20%	-0.08	0.54	58%	95	2.14	0.04
Enet	4.54%	6.90%	0.66	9.10%	0.11	0.08	52%	95	1.85	0.07
OLS	4.34%	7.12%	0.61	11.20%	-0.07	0.41	60%	95	1.72	0.09
Step	3.70%	7.33%	0.50	11.60%	-0.10	0.73	58%	95	1.42	0.16

Figure 4.4. *Backtest results, MSCI USA (source: Macquarie Quantitative Strategy, January 2019. The backtest is based on five equally weighted baskets)*

The results indicate that OLS and stepwise regression, our linear benchmarks, are outperformed by all machine learning alternatives in terms of risk-adjusted returns (IR). Tree-based models are the best performers, with IRs of 1.13 (gradient-boosted machine) and 1.39 (random forest) which are about double the IR obtained by the OLS model.

The Alpha model performs well, as demonstrated by its IR of 0.82 which places it close to neural networks in the performance rankings. As a summary of the predictive ability of our proposed machine models, we can focus on the IR of the combined nonlinear model (ensemble) which, at 1.18, is one of the highest.

While it is hard to measure downside risk for a relatively short sample period, we note that the machine learning models improve on the Alpha

model in terms of maximum drawdown, skewness and volatility. A possible explanation for this result is that the Alpha model is affected by momentum drawdowns to a larger extent than the machine learning counterparts, which also load on a variety of technical factors.

To better assess the stock picking skills of the machine learning models in our sample period, we compare them with the returns of long–short Fama–French (1992, 2015) (FF) factors. We use returns to equally weighted portfolios constructed by double sorting stocks based on their size and then on the relevant signal. In particular, for all factors except size, we focus on large firms (i.e. those above the median market value of NYSE stocks)[9]. Furthermore, we also report results for the standard size factor.

Figure 4.5 shows the performance of each factor plus a combination of the five signals. The results show that value (high minus low book-to-price) and size (small minus big) are the worst performing factors from January 2011 to November 2018, with IRs of -0.28 and -0.49, respectively. The other factors (investment, momentum and profitability) display a positive IR, but their positive returns are not high enough to yield a positive IR for the combined strategy (combination). The latter would have generated a negative IR (-0.04) during our backtesting period. Consequently, since these factors represent the broad investment styles covered by our features, the machine learning models are up against a very difficult task in the USA.

Factor	Mean return	Volatility	IR	Max drawdown	Skewness	Excess Kurtosis	Hit rate	Number of obs	t-Test	p value
Book-to-Price	-2.06%	7.49%	-0.28	25.30%	0.97	2.26	42%	94	-0.77	0.44
Investment	0.65%	5.51%	0.12	13.80%	0.21	-0.18	46%	94	0.33	0.74
Momentum	3.48%	11.40%	0.31	20.10%	0.22	1.22	50%	94	0.86	0.39
Profitability	1.20%	5.83%	0.21	11.20%	0.26	1.12	55%	94	0.58	0.56
Size	-3.98%	8.17%	-0.49	29.30%	0.18	-0.31	40%	94	-1.36	0.18
Combination	-0.14%	3.14%	-0.04	8.40%	0.20	0.63	50%	94	-0.13	0.90

Figure 4.5. *Performance of Fama–French factors in the USA, January 2011–November 2018 (source: Kenneth French website, Macquarie Quantitative Strategy, January 2019. For book-to-price, investment, momentum and profitability, we use the large cap version of the factors. Portfolios are equally weighted)*

9 The large cap, equally weighted versions of the FF factors are more relevant because our universe has fewer constituents than the one used for the standard value, operating profitability and investment factors and our portfolios are equally weighted. As Asness and Frazzini (2013) point out, factor performance can differ significantly between large and small caps.

Figure 4.6 reports the cumulative returns of the strategy. Unsurprisingly, the time pattern of returns is very similar across different strategies. This is due to the fact that we use the same building blocks for all models. OLS is clearly the worst performer, and most of the gap in cumulative return is realized in the first two years of the out-of-sample period and between 2015 and 2016.

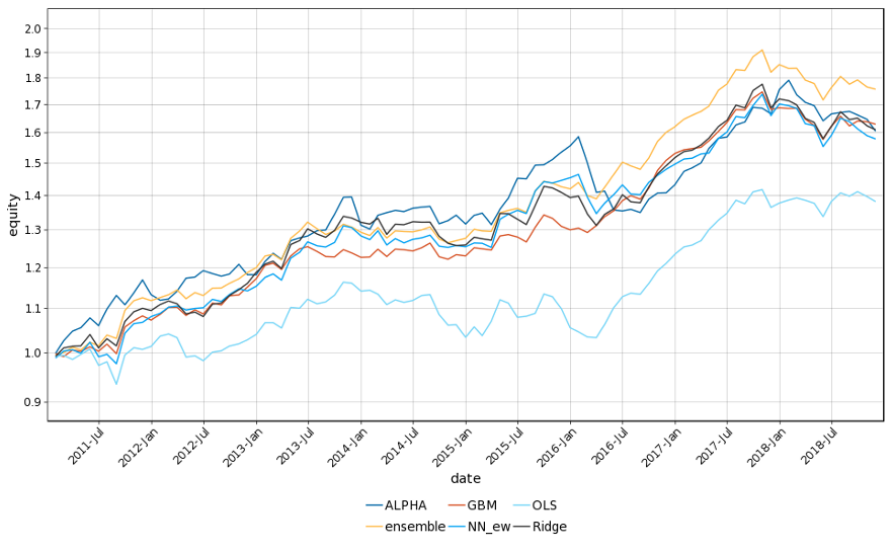


Figure 4.6. *Simulated performance over time, MSCI USA (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Our Alpha model performs in line with the machine learning-driven alternatives in terms of total return; however, it displays higher volatility, particularly at a time of sharp style rotation between 2015 and 2016 when value styles went through a bout of dramatic underperformance. The combination of nonlinear models (ensemble) seems to slightly outperform all alternatives by limiting the drawdown in early 2016.

4.5.3. *Returns correlation with Alpha model*

Figure 4.7 reports the returns correlations between the Macquarie Alpha model and each of the main machine learning alternatives. We also report

the rank correlation between the alpha signals. Neural networks display the highest correlations.

	GBM	RF	NN
TS corr	27.61%	40.96%	52.51%
Rank corr	28.09%	26.25%	46.68%

Figure 4.7. *Correlation between machine learning models and Macquarie Alpha model (source: Macquarie Quantitative Strategy, January 2019)*

4.5.4. *Active returns by basket*

In Figure 4.8, we plot the average excess returns of quintile baskets in excess of *equally weighted* benchmarks. The use of the standard cap-weighted indices would be misleading as excess returns would be affected by the differences in portfolio construction, which would tend to favor the equally weighted baskets. Hence, we use an equally weighted version of the relevant MSCI index. Confidence bands are based on the t-distribution and can be used to assess the significance of the outperformance of our proposed strategies.

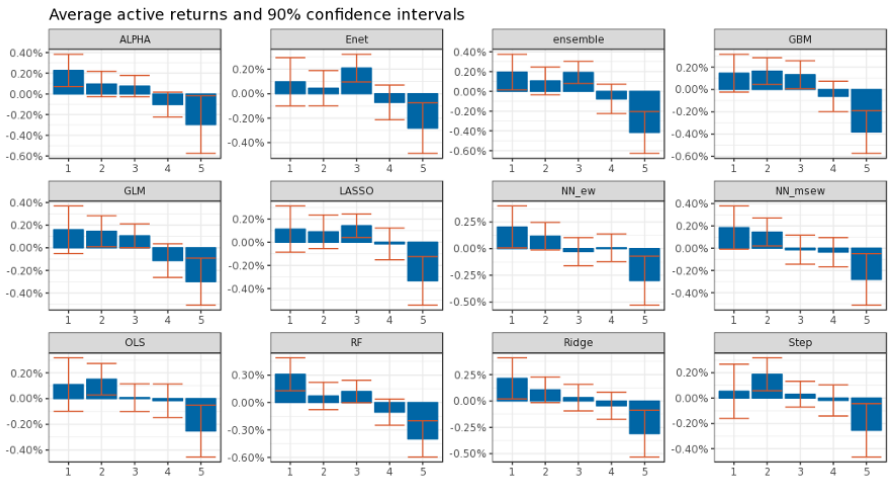


Figure 4.8. *Active returns, MSCI USA (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The signs of the active returns are all as expected: quintile 1 (most attractive stocks) invariably outperforms the benchmark, while quintile 5 underperforms in all cases. Some of the strategies display a very asymmetric pattern: for example, gradient-boosted machine and OLS both rely on the short side to generate outperformance. Confidence bands also suggest that the return to the long basket is not significantly higher than the benchmark for both models, as the 90% interval includes the value zero. Models such as our Alpha model and random forests are better behaved as they display significant active returns both on the long and the short side.

4.5.5. *Calibrated hyperparameters and model complexity*

How are the models training in sample and what kind of solution do they find? How complex are the model architectures? Would we be able to understand in advance whether a model has a poor fit? To answer these questions, in this section, we plot some of the hyperparameters that characterize the architectures of our models. We focus on trees because the hyperparameters are directly related to the complexity of the models, and we do not report complexity parameters for penalized linear models.

We do not report the hyperparameters for the neural network because they do not have the same intuitive interpretation as in the case of tree-based models. An interesting extension would be to calibrate the depth and the width of the network and analyze how the architecture changes over time.

Each hyperparameter is the result of a grid search performed only once per year, i.e. at the beginning of the year, and using the most recent eight years as validation. In Figure 4.9, we report the average tree depth, the number of trees, the number of distinct features and the parameter m (mtries) for random forest models. The number of distinct features is the average number of distinct covariates that goes into a tree of the forest, i.e. because we select m variables at each level of splits, one tree can potentially split by the same feature multiple times. Hence, to calculate this, we consider each tree of the forest and calculate the number of distinct features that were used in that tree. Then, we average this number across all trees.

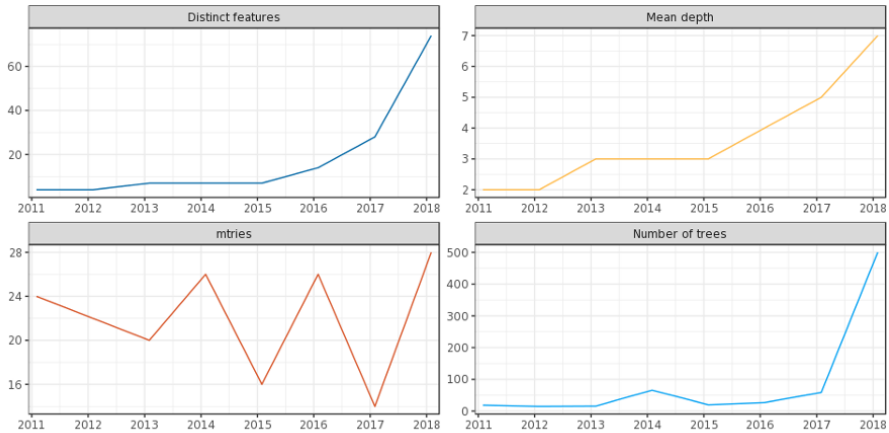


Figure 4.9. *Hyperparameters: random forest (source: Macquarie Quantitative Strategy, January 2019)*

In line with Gu *et al*. (2018), we find that tree complexity increases over time, with an average tree depth increasing from 2 (in January 2011) to 7 (in January 2018). Also, the number of distinct features entering the trees increases over time. The optimal hyperparameters vary over time, but the bounds we used in the calibration turned out to be sufficiently wide to accommodate those variations. Only at the last re-estimation date, in January 2018, has the random forest grown the maximum number of allowed trees, i.e. it has attained the upper bound of the parameter governing the number of trees (500). As this is substantially higher than the optimal number of trees grown in the rest of the sample (always below 100), this might signal that the model is struggling to find a solution and keeps adding trees since the tolerance parameter is very low.

Figure 4.10 plots the number of distinct features, the number of trees and the shrinkage factor for the gradient-boosted machine at each recalibration date. It is worth noting that, even though we only use stock-specific features, we end up with a very similar number of distinct features entering the trees, as in Gu *et al*. (2018), who added interactions with macro variables, ending up using approximately 1,000 features, 10 times more than ours. This implies that interactions between stock-specific factors and macro variables are not important features.

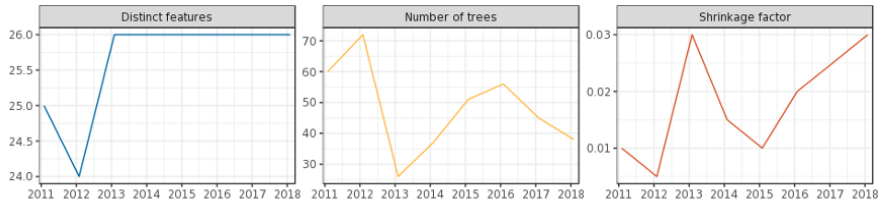


Figure 4.10. *Hyperparameters: gradient-boosted machine (source: Macquarie Quantitative Strategy, January 2019)*

4.5.6. *Variable importance*

The results for our random forest model are presented in Figure 4.11. Darker shades of blue indicate higher variable importance and the figures are aggregated by broad style family. As has been documented in the academic literature, price-driven predictors tend to dominate consistently over time. Technical indicators, in particular, rank at the top in terms of variable importance. The model also tends to select price momentum indicators and, to a lesser extent, valuation ratios in the "defensive value" category such as dividend yield.

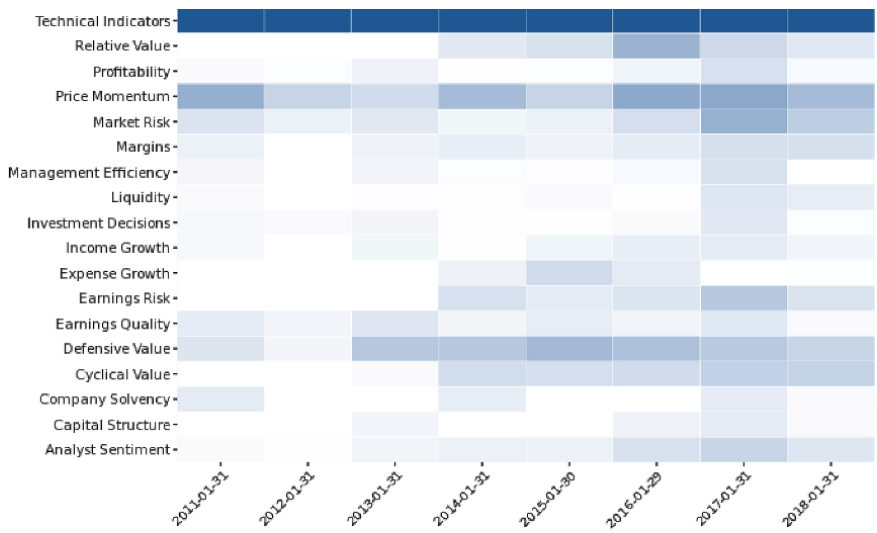


Figure 4.11. *Variable importance: random forest model (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Boosted regression tree models produce very similar results, albeit somewhat less extreme (Figure 4.12). Market risk, another category dominated by price-driven characteristics, also features among the most important ones in this case. Along with defensive value metrics, the fundamental characteristics are represented by measures of the strength of a company's margins.

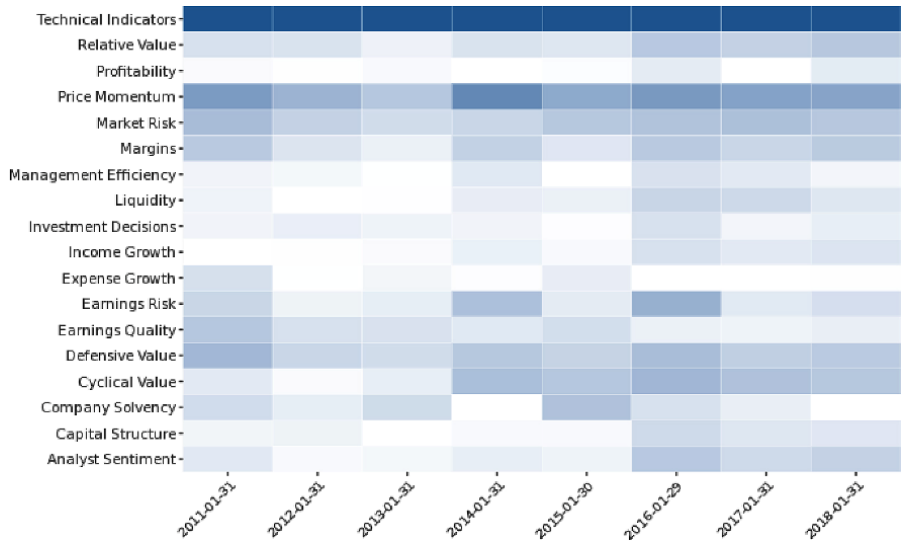


Figure 4.12. *Variable importance: gradient-boosted machine (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The picture is more blurred, unsurprisingly, for the neural network model (Figure 4.13). In this case, technical indicators are still predominant, but the model does load on fundamental characteristics such as defensive value ratios, profitability indicators and measures of expense growth. Variable importance seems to vary over time more markedly than for the other models, and towards the end of the sample period it appeared to be concentrated on technical indicators.

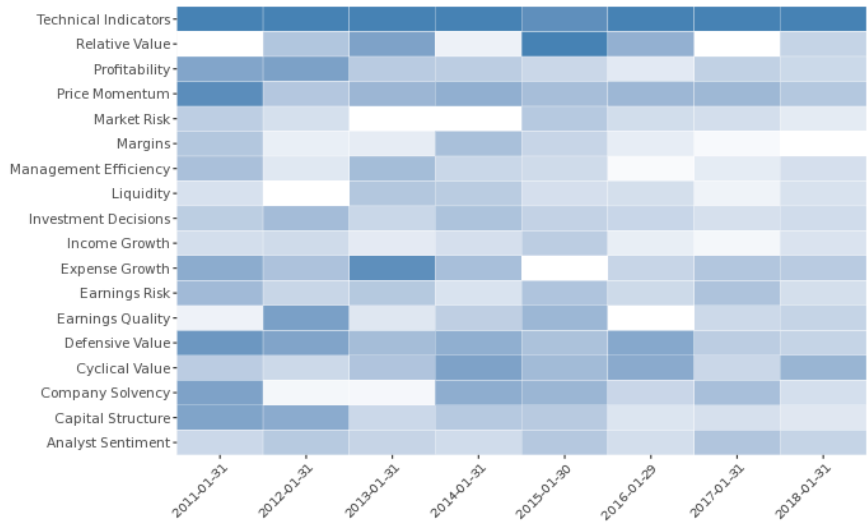


Figure 4.13. *Variable importance: neural network (source: Macquarie Quantitative Strategy, January 2019. The plot refers to the model obtained by equally weighting five different networks obtained by setting five different random seeds (NN_ew)). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.6. Predicting European stock returns

In this section, we analyze the stock returns predictions for the constituents of MSCI Europe from 1995 to 2018.

4.6.1. *Information coefficients*

We plot average ICs for the European models in Figure 4.14. In contrast to the US results in Figure 4.3, the traditional linear predictors (OLS and step) perform well compared to tree-based models (gradient-boosted machine and random forest) and LASSO. Here, the average out-of-sample (test) IC of OLS is above 4%, slightly higher than the gradient-boosted machine and random forest. The best performing models are neural networks (with an average IC above 5%) and ridge regression. However, when ICs are so tightly clustered, it is difficult to draw clear conclusions as they are a noisy measure of predictive accuracy.

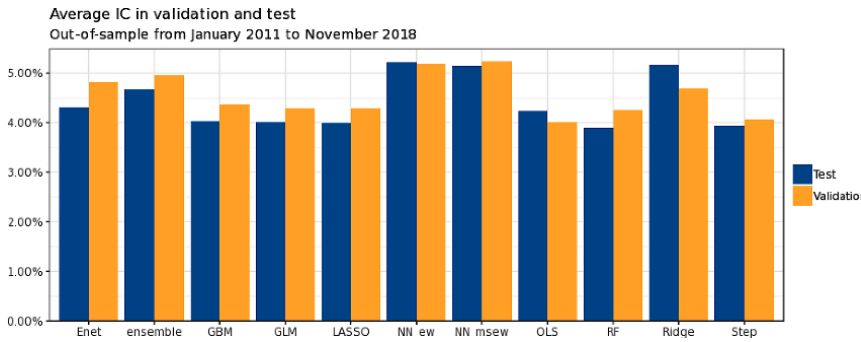


Figure 4.14. *Information coefficients, MSCI Europe (source: Macquarie Quantitative Strategy, January 2019)*

Comparing ICs on an annual basis results in a rather mixed picture (Figure 4.15). In validation, the ranking sees our neural network model promise the highest forecasting accuracy. However, in the test subsample, the performance of the equally weighted neural network is not as clearly dominant.

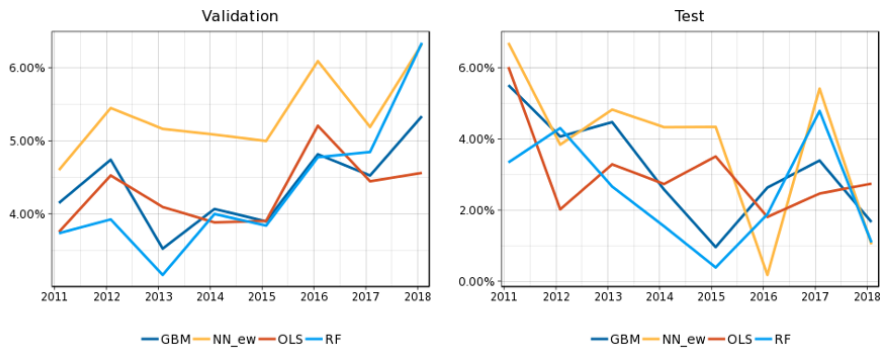


Figure 4.15. *Average IC in validation (eight years) and test (one year), MSCI Europe (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.6.2. *Long–short strategy*

An alternative way to compare the different methods consists of running a simple backtest with the same setup used to generate the results in Figure 4.4 for US stocks, i.e. equally weighted quintile portfolios. Figure 4.16

shows the results, where strategies are ranked by their IR. The good performance of neural networks and ridge regression, as we have seen earlier in Figures 4.14 and 4.15, is confirmed by the backtest results which rank them at the top. Gradient-boosted machine, despite the disappointing average IC, does outperform OLS based on risk-adjusted returns. However, the other tree-based model (random forest) clearly struggles to outperform the standard OLS approach in this dataset.

Model	Mean return	Volatility	IR	Max drawdown	Skewness	Excess Kurtosis	Hit rate	Number of obs	t-Test	p value
Ridge	11.30%	7.55%	1.49	9.40%	-0.15	1.44	74%	95	4.20	0.00
NN_ew	11.80%	8.65%	1.36	10.40%	-0.23	1.48	67%	95	3.83	0.00
ensemble	10.00%	7.61%	1.32	8.70%	0.20	1.23	66%	95	3.70	0.00
GBM	8.84%	6.82%	1.30	7.30%	0.17	0.53	67%	95	3.64	0.00
NN_msew	11.30%	8.69%	1.30	9.50%	-0.16	1.47	67%	95	3.66	0.00
Enet	8.97%	7.89%	1.14	11.00%	0.06	1.82	66%	95	3.20	0.00
OLS	7.93%	7.21%	1.10	9.80%	-0.42	1.78	70%	95	3.09	0.00
GLM	8.80%	8.12%	1.08	10.80%	0.07	1.97	64%	95	3.05	0.00
LASSO	8.46%	7.98%	1.06	10.60%	0.07	1.85	63%	95	2.98	0.00
Step	7.82%	7.63%	1.02	11.30%	0.03	1.49	61%	95	2.88	0.01
RF	7.57%	7.54%	1.00	6.50%	0.74	1.40	57%	95	2.83	0.01
ALPHA	9.73%	10.10%	0.96	10.40%	-0.30	0.97	63%	95	2.71	0.01

Figure 4.16. *Backtest result, MSCI Europe (source: Macquarie Quantitative Strategy, January 2019. The backtest is based on five equally weighted baskets)*

The magnitude of IRs is large, bearing in mind that the portfolios are not optimized and the rebalancing frequency is monthly. Our Alpha model, despite an IR of almost one, is outperformed by all the other predictive models. Stepwise regression only manages to beat random forest, but it underperforms all the other models. A combination of the nonlinear predictors (ensemble) results in a very high IR (1.32).

The main driver of the underperformance of our Alpha model is risk: the equally weighted portfolio displays a volatility of more than 10% over the period, while the machine learning-driven alternatives have volatilities ranging between 6.82% (gradient-boosted machine) and 8.69% (MSE-weighted neural network).

Finally, we note that the Alpha model and OLS display the most negative skewness, in line with our findings for the US model.

Following our analysis of US returns, we compare the performance of our machine learning models with the returns of FF factors in developed Europe during the same sample period. Figure 4.17 suggests that FF factors worked

better in Europe than in the USA over that period, which explains why the performance of *all* models is stronger in Europe. A combined strategy of value, investment, momentum, profitability and size would have generated an IR of 0.48 in the backtesting period. This evidence points to the importance of training machine learning models using the correct inputs. Note that a simple ridge estimator, which takes into account the strong collinearity among features but does not try to perform any variable selection, would have outperformed the ensemble of machine learning models in Europe.

Factor	Mean return	Volatility	IR	Max drawdown	Skewness	Excess Kurtosis	Hit rate	Number of obs	t-Test	p value
Book-to-Price	-4.71%	9.52%	-0.49	39.20%	-0.01	-0.53	46%	94	-1.38	0.17
Investment	-0.70%	5.00%	-0.14	13.50%	-0.28	-0.30	54%	94	-0.39	0.70
Momentum	6.61%	10.40%	0.64	16.40%	-0.67	3.47	57%	94	1.79	0.08
Profitability	2.86%	6.03%	0.47	12.30%	-0.10	-0.10	55%	94	1.33	0.19
Size	1.50%	5.60%	0.27	10.30%	-0.09	-0.06	51%	94	0.75	0.46
Combination	1.11%	2.34%	0.48	7.20%	-0.49	1.04	53%	94	1.33	0.19

Figure 4.17. *Performance of Fama–French factors in Europe, January 2011–November 2018 (source: Kenneth French website, Macquarie Quantitative Strategy, January 2019. For book-to-price, investment, momentum and profitability, we use the large cap version of the factors. Portfolios are equally weighted)*

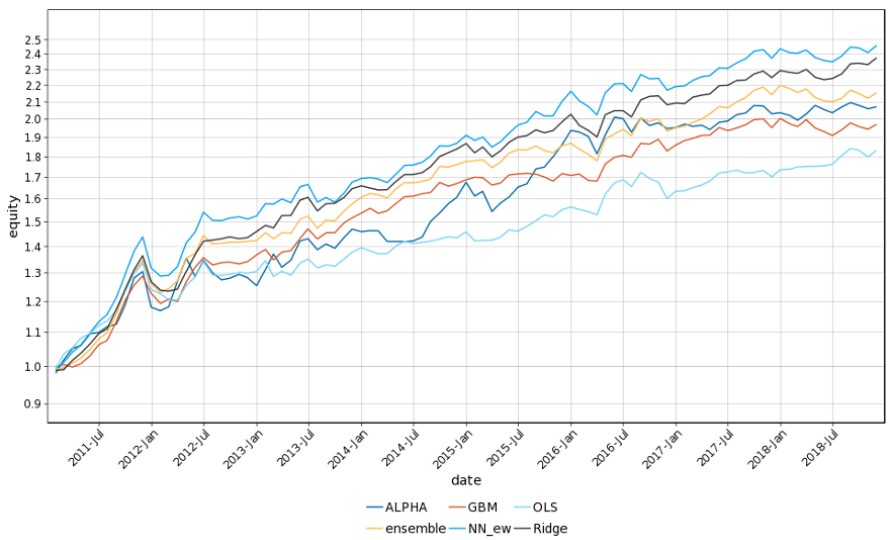


Figure 4.18. *Simulated performance over time, MSCI Europe (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The higher volatility generated by our Alpha model is apparent in the cumulative return plot of Figure 4.18. OLS seems to underperform the other models, like in the analysis of US data, at the beginning of the sample period, while, in 2018, it recovers some of the lost ground.

4.6.3. *Returns correlation with Alpha model*

Figure 4.19 reports the returns correlations between the Macquarie Alpha model and each of the main machine learning alternatives. We also report the rank correlation between the alpha signals. Neural networks display the highest correlations.

	GBM	RF	NN
TS corr	46.52%	60.99%	75.88%
Rank corr	33.88%	31.66%	48.84%

Figure 4.19. *Correlation between machine learning models and Macquarie Alpha model, MSCI Europe (source: Macquarie Quantitative Strategy, January 2019)*

4.6.4. *Active returns by basket*

We assess the monotonicity of the relation between alpha signals and subsequent returns in Figure 4.20. Compared to the results for the US universe, the patterns in the figure are even more encouraging, in the sense that most multifactor signals display a monotonic negative relation.

In most cases, both the long and the short side generate significant excess returns on an equally weighted benchmark of MSCI Europe constituents. The magnitude of returns is higher for neural networks and the Alpha model, but lower for OLS and random forest.

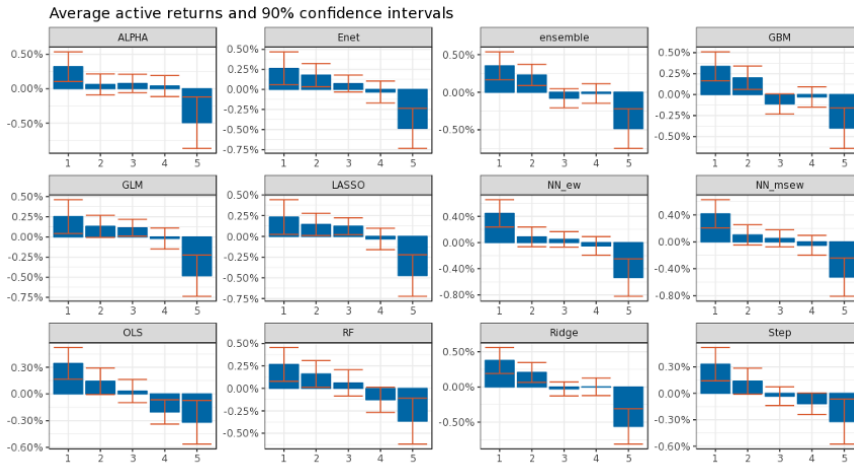


Figure 4.20. *Active returns, MSCI Europe (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.6.5. *Calibrated hyperparameters and model complexity*

We plot the calibrated hyperparameters of the random forest model over time in Figure 4.21. The evolution of the hyperparameters is similar to the one found for the US sample (see Figure 4.9). In particular, model complexity tends to increase as the training window expands. This is in line with the empirical findings reported by Gu *et al*. (2018).

As shown in Figure 4.22, the gradient-boosted machine displays a more stable structure, confirming the result obtained in the US universe. The maximum tree depth for the gradient-boosted machine is bounded at five, so it could be that the trees are trying to grow more; however, since they are limited, they start to combine more trees with lower learning rate (shrinkage factor).

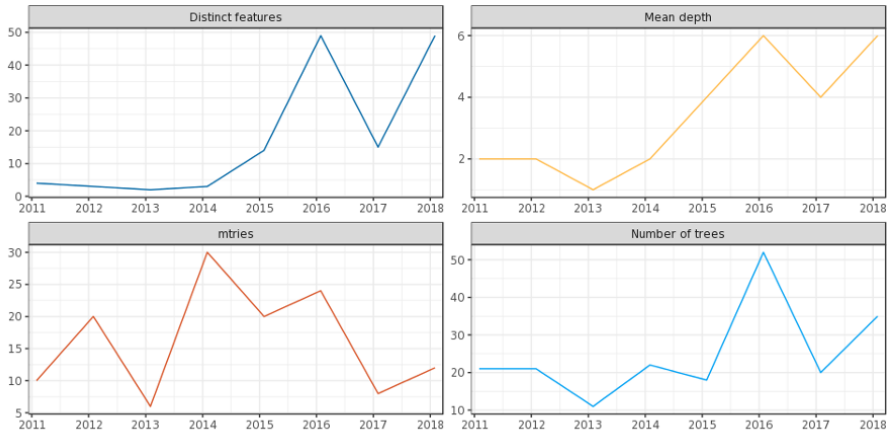


Figure 4.21. *Hyperparameters: random forest (source: Macquarie Quantitative Strategy, January 2019)*

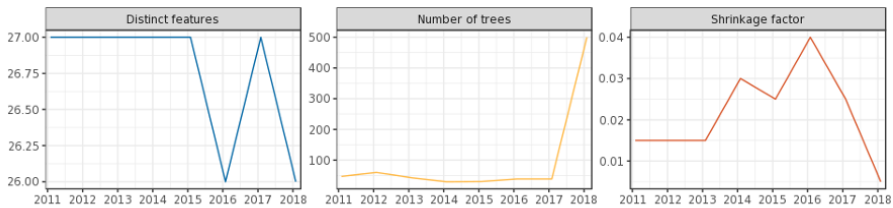


Figure 4.22. *Hyperparameters: gradient-boosted machine (source: Macquarie Quantitative Strategy, January 2019)*

4.6.6. *Variable importance*

In terms of variable importance, the random forest models estimated on US data (Figure 4.11) and European data (Figure 4.23) produce very consistent results. Price-driven signals are again the main features selected by the model.

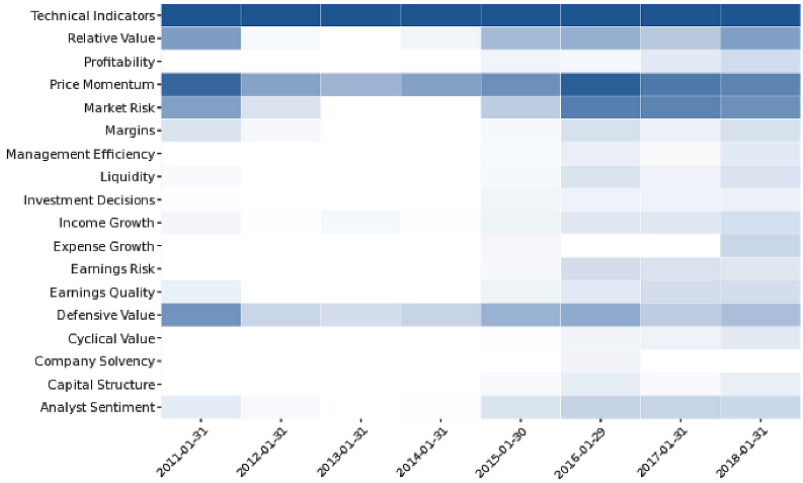


Figure 4.23. *Variable importance: random forest model (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

A similar conclusion can be reached by inspecting the variable importance plot for the gradient-boosted machine (Figure 4.24). The market risk category appears to be even more prominent in the European model than in the US model.

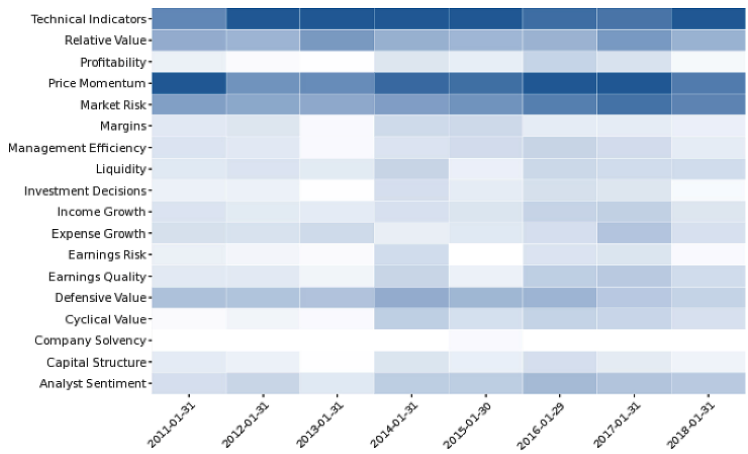


Figure 4.24. *Variable importance: gradient-boosted machine (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The results for our neural network model (Figure 4.25) are also similar to those found in the analysis for the USA. The matrix is less sparse than for gradient-boosted machine or random forest, but price-driven signals still play the main role.

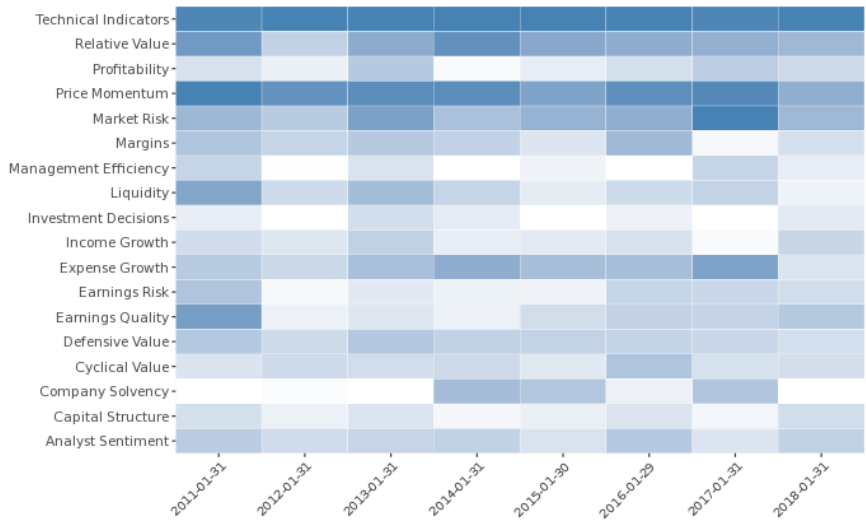


Figure 4.25. *Variable importance: neural network (source: Macquarie Quantitative Strategy, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.7. The impact of transaction costs

So far we have tested the predictive power of our proposed machine learning models by running simple equally weighted backtests. In this section, we use the Axioma portfolio optimization system to simulate more realistic trading strategies and gauge the impact of transaction costs on performance.

4.7.1. *Optimized strategies for European stocks*

We choose a fairly standard optimization setup to select a long–short portfolio that is mean variance efficient subject to certain realistic

constraints. In particular, we impose a maximum risk level of 5% (annualized) and maximize expected returns using the alphas from each model in turn as an input. The portfolio is fully collateralized (i.e. if the amount of collateral is 100, then we open long and short positions worth a total of 100 on each side).

The optimization problem also includes a series of constraints that are common in quantitative trading strategies:

– net sector exposure is limited within -5 and 5% of the net portfolio value using level 1 GICS sectors;

– country exposures are also bound within -5 and 5%;

– currency bets are restricted within the -1% to 1% bound;

– individual stock weights are capped at 2%. We also impose a threshold of 0.50% to avoid very small positions;

– when rebalancing, we limit the amount that can be traded in a security to 10% of its 20-day average daily volume (ADV).

The last constraint is meant to limit market impact by reducing the trading footprint of the strategy in the market. We assume a starting capital of EUR 100m in January 2011. In order to gauge the impact of trading costs, we impose a one-way cost of 10bp, which is in line with typical trading costs faced by a large institutional investor, as documented by Frazzini *et al*. (2018)[10].

Figure 4.26 shows the results of our simulation. The main conclusions of our analysis can be summarized as follows. In terms of risk-adjusted returns *before transaction costs* ("gross IR"), our machine learning models outperform the traditional alpha model, except for random forest. This result is in line with our non-optimized backtest in the previous sections. The gap in information ratios between the Alpha model and machine learning models is sizeable: the former has a (still very respectable) IR of 1.80, while the combined machine learning model (ensemble) reaches 2.16.

10 Under this assumption, to liquidate the long portfolio and substitute all holdings with new positions would cost 20bp of its value.

	Alpha model	GBM	NN_ew	RF	Ensemble
Gross return (aritm.)	11.8%	10.1%	12.9%	8.1%	11.9%
Net return (aritm.)	10.3%	6.8%	10.1%	5.0%	8.9%
Gross return (geom.)	12.2%	10.4%	13.5%	8.3%	12.4%
Net return (geom.)	10.6%	6.9%	10.4%	4.9%	9.1%
Volatility	6.7%	5.3%	6.5%	6.0%	5.7%
Gross IR	1.80	1.96	2.08	1.37	2.16
Net IR	1.57	1.30	1.60	0.81	1.58
Average turnover	1.19	2.74	2.33	2.68	2.59
Average no. stocks (L)	59.2	63.1	65.6	62.4	64.4
Average no. stocks (S)	60.3	63.2	63.4	62.8	63.5
Max drawdown	5.9%	7.9%	6.2%	8.5%	3.6%

Figure 4.26. *Backtest results for European optimized strategies (no turnover constraint) (source: Macquarie Quantitative Strategy, Axioma, January 2019. Transaction costs are assumed to be 10bp one-way. Returns are computed in euros)*

However, the new models deliver such strong performance by producing a large amount of trading, as indicated by the average monthly turnover that is typically more than double that of the Alpha model (about 259% vs. 119%). We argue that this result is due to the strong exposure of such strategies to fast moving signals such as technical indicators. As a result, our assumption of proportional transaction costs impacts the machine learning models much more than the benchmark. Net returns are about 3% lower than gross returns for the new model, while the gap is closer to 1.5% for the benchmark.

It is worth mentioning that turnover is calculated as a total of long and short sides and expressed as a percentage of the collateral value. Hence, the maximum turnover that can be generated when rebalancing the portfolio is 400%.

The net information ratios shown in the table suggest that, despite the strong impact of transaction costs, the machine learning models do deliver a level of performance that is comparable to our Alpha model. In fact, the net IR for our combined machine learning model is still slightly higher than that of our benchmark (1.58 vs. 1.57). Of the three proposed models, neural network has both the lowest turnover and the highest net IR (1.60).

It is clear from the table that our machine learning models tend to result in a higher number of portfolio positions, both on the long and the short side. This seems to generate diversification benefits that result in an out-of-sample volatility that is closer to the target (5%) than the volatility of the Alpha model.

To investigate the impact of turnover on the performance of our strategies, we impose a further constraint on the optimization: we impose a cap on the total portfolio turnover. Figure 4.27 summarizes the results by focusing on the combined machine learning model (ensemble) and the benchmark (Alpha model). The dashed lines represent the relation between gross risk-adjusted returns and turnover levels. Before transaction costs, unsurprisingly, a tighter constraint on the amount of trading results in a deterioration in performance. If we impose a maximum monthly turnover of 80% (furthest point to the left of the chart) then the gross IRs of the two strategies are both approximately 1.60. For looser turnover constraints (120% and 160% limits), the machine learning model delivers superior performance. What is the impact of transaction costs in this case?

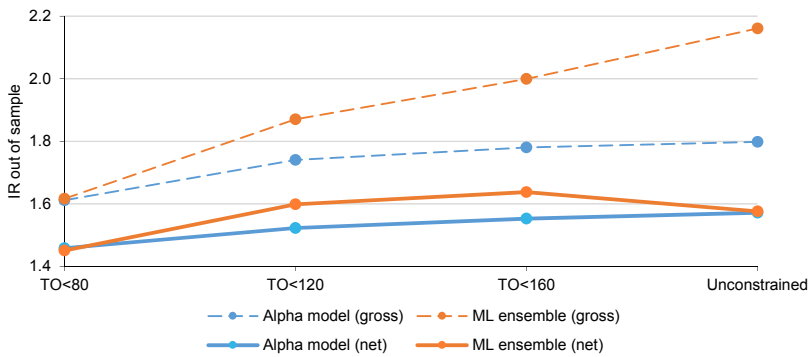


Figure 4.27. *Effect of turnover constraints pre- and post-transaction costs. ML = machine learning (source: Macquarie Quantitative Strategy, Axioma, January 2019. Turnover is measured as a monthly percentage and is the sum of two-way turnover for both sides). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The solid lines in Figure 4.27 trace the IR of the two portfolios' net of transaction costs. As expected, the impact of trading costs is much higher for the "ensemble" strategy. Nevertheless, our machine learning signals do not underperform the Alpha model even after transaction costs. The risk-adjusted

performance results are very close, with the new models having a slight advantage on the benchmark.

Figure 4.28 further illustrates the results for the case with a turnover limit of 160% by plotting the cumulative total return of our strategies. The ensemble strategy has a comparable performance to the benchmark over the full period; however, its return pattern over time is more regular, which explains the lower *ex post* risk. Random forest and regression trees not only display lower absolute returns but also have much lower levels of risk, closer to the target of 5%. The overall behavior of the strategies is very similar, which is not surprising given that the raw signals are the same. For example, 2018 turned out to be a challenging year for all models.

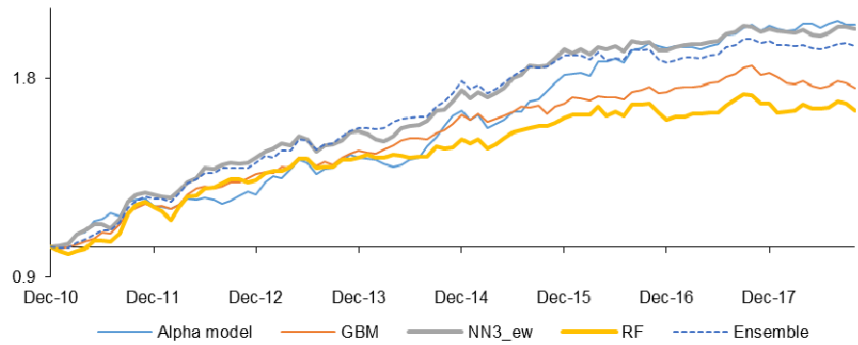


Figure 4.28. *Cumulative total return after costs with turnover limit (source: Macquarie Quantitative Strategy, Axioma, January 2019. Turnover is capped at 160% per month. Total return is plotted on a logarithmic scale). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.7.2. *Optimized strategies for US stocks*

In this section, we repeat the analysis of the impact of transaction costs using US data. The optimization problem is identical to the one described in section 4.7.1, except that:

– we drop currency and country constraints which are not relevant in this context;

– we assume an initial capital of USD 100m;

– individual stock weights are capped more tightly at 1.5% given that the US index contains a higher number of constituents compared to the European index.

The results are shown in Figure 4.29. The Macquarie Alpha model would have delivered strong performance, as indicated by the IR of 1.11 gross of trading costs. Factoring in an average turnover of almost 140% per month, however, results in a net IR of just 0.87. The impact of transaction costs on annualized returns is 1.7%.

	Alpha model	GBM	NN_ew	RF	Ensemble
Gross return (aritm.)	7.9%	6.6%	5.2%	7.1%	6.8%
Net return (aritm.)	6.3%	3.0%	2.0%	3.4%	3.2%
Gross return (geom.)	7.9%	6.6%	5.1%	7.2%	6.8%
Net return (geom.)	6.2%	2.9%	1.9%	3.3%	3.1%
Volatility	7.1%	5.1%	6.2%	5.6%	5.4%
Gross IR	1.11	1.32	0.81	1.29	1.26
Net IR	0.87	0.57	0.30	0.59	0.57
Average turnover	1.39	3.05	2.64	3.13	3.02
Average no. stocks (L)	70.3	69.1	70.4	69.8	69.6
Average no. stocks (S)	69.6	68.4	68.4	68.4	68.5
Max drawdown	11.9%	10.1%	14.3%	11.8%	11.5%

Figure 4.29. *Backtest results for US optimized strategies (no turnover constraint) (source: Macquarie Quantitative Strategy, Axioma, January 2019)*

The combination of machine learning models ("ensemble") tends to generate even higher turnover due to its exposure to technical indicators and momentum signals. While it outperforms the Alpha model before transaction costs (its gross IR being 1.26), it displays a lower net IR of 0.57. The gap between gross and net annualized returns is more than double at 3.6%.

Among the different machine learning models, tree-based techniques deliver the best performance as expected. They also generate the highest turnover.

By curbing the amount of trading, it is possible to generate much more attractive risk-adjusted net returns, as shown in Figure 4.30. As in the European section, we impose a constraint on the total portfolio turnover.

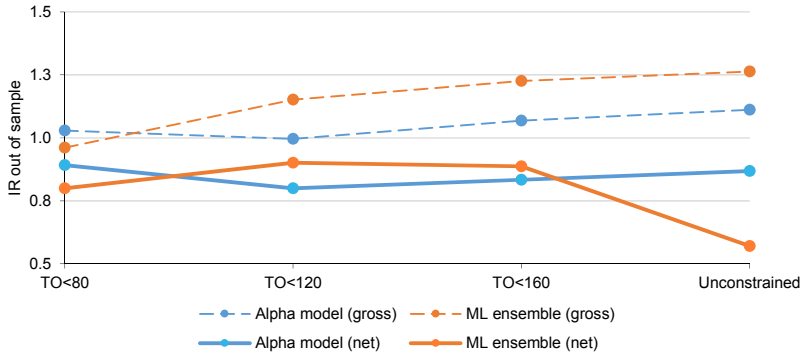


Figure 4.30. *Effect of turnover constraints pre- and post-transaction costs (US data). ML = machine learning (source: Macquarie Quantitative Strategy, Axioma, January 2019). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The dashed lines plot the relation between gross IR and turnover limits. As expected, the performance of our machine learning model deteriorates as we gradually tighten the constraint from 160% to 80%. The net IR, however, jumps from 0.57 (unconstrained case) to 0.9 (120% limit), as the solid orange line shows. The Alpha model and machine learning ensemble produce similar levels of risk-adjusted performance after transaction costs for all three values of the turnover constraint. Both at 120% and 160% turnover, the combined machine learning model outperforms mildly.

The cumulative total return plot in Figure 4.31 suggests that the machine learning ensemble and the Alpha model would have displayed a similar pattern of returns over time but the latter would have suffered from more pronounced swings, particularly due to the momentum rally in 2015 and the subsequent crash.

The analysis of optimized strategies in this section leads to some encouraging results. Despite the fact that the optimization does not explicitly target transaction costs, the machine learning models were able to achieve similar levels of performance net of trading costs to our traditional factor model. We would like to emphasize that the Macquarie Alpha model is a challenging benchmark to measure any strategy against, given its strong out-of-sample performance. These findings suggest that the strong exposure of the machine learning models to price momentum and technical factors does not prevent our proposed models from generating significant outperformance

at realistic levels of transaction costs. Further robustness checks that confirm this result are available from the authors.

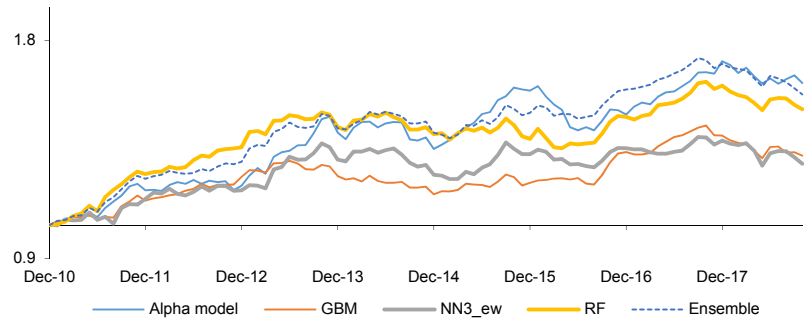


Figure 4.31. *Cumulative total return after costs with turnover limit (US data) (source: Macquarie Quantitative Strategy, Axioma, January 2019. Turnover is capped at 120% per month. Total return is plotted on a logarithmic scale). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

4.8. Conclusion

In this chapter, we have investigated the potential application of machine learning methods to the problem of predicting stock returns using a large set of firm characteristics. Our goal was to test in a liquid universe the results obtained by several recent academic studies, extend the analysis to European equities and gauge the impact of realistic transaction costs on performance.

The main empirical results can be summarized as follows. First, machine learning techniques have strong potential and, even when compared to a proprietary model with a successful out-of-sample record like our Macquarie Quant Alpha model, have enough predictive power to generate attractive risk return ratios. In particular, we conclude that a combination ("ensemble") of various machine learning models would have outperformed our Alpha model both in the USA and Europe, reaching similar average returns but with lower volatility.

Moreover, we find (in line with the empirical results reported in academic papers) that price momentum factors and technical indicators dominate the optimal signal combination. Neither company fundamentals based on balance sheet information nor other slow-moving signals seem to play an important

role in the estimated models. To summarize, price information that is updated at monthly frequency appears to be the main source of predictability.

However, this does not imply that the strategy's alpha is eroded by transaction costs. In a realistic optimization setup, we showed that, at a reasonable level of trading costs, the net risk-adjusted performance of the machine learning-driven strategies is still attractive. Compared to our Alpha model, the proposed machine learning strategies have a similar pattern of returns over time, but they tend to be smoother and are less pronounced downside. This lower level of risk is caused, at least partly, by the fact that machine learning models tend to be less exposed to price momentum than the Alpha model during the momentum rally and the subsequent crash in 2015–2016.

Our results also indicate that the number of selected predictors and relative weight change significantly over time. In particular, the selection broadens towards the end of the sample period as our expanding window increases.

Finally, a very important lesson we learned from this exercise is that the default hyperparameters values and typical ranges used in the available off-the-shelf packages turned out to be unsuitable for our purposes. The reason for this discrepancy is arguably related to the peculiar nature of financial data that makes it very different from the data typically used in a successful machine learning application. In particular, the lower signal-to-noise ratio affects the optimal choice of hyperparameters such as learning rates and regularization coefficients. As a result, calibrating the hyperparameters required a considerable amount of effort.

Further research can be developed in several directions. First, the machine learning models can be trained on a wider universe (e.g. MSCI World) and then can be used to predict a smaller set of stocks (e.g. MSCI USA or MSCI Europe). If the performance of the models is positively related to the amount of data available, this approach should help the models to learn. Region-specific dummy variables can be easily added to the set of features to take into account possible heterogeneity of the relationship between firm characteristics and stock returns. Second, the models can be trained exclusively with price momentum factors and technical indicators to create a short horizon trading model. In this case, we could even use data at a weekly or daily frequency. Third, model weights can be updated more

frequently than in this chapter (where we keep them fixed for one year). In particular, the models can be trained before each prediction, without altering the hyperparameters. Fourth, ensembles of models can be constructed more efficiently, for example, using the stacked ensemble functions provided by H2O or by simply increasing the number of elements in the neural network ensemble. Fifth, the structure of the neural networks can also be calibrated, adding the number of neurons and the number of layers to the hyperparameter grid. Finally, various unsupervised methods can be used to first group the features into styles before feeding them to the supervised models.

4.9. References

Abe M. and Nakayama H. (2018). Deep learning for forecasting stock returns in the cross-section. In Phung D., Tseng V.S., Webb G.I., Ho B., Ganji M. and Rashidi L. (eds). *Advances in Knowledge Discovery and Data Mining*, Springer, New York.

Asness C. and Frazzini A. (2013). The devil in HML's details. *Journal of Portfolio Management*, 39, 49–68.

Asness C., Frazzini A., Israel R., Moskowitz T.J. and Pedersen L.H. (2018). Size matters, if you control your junk. *Journal of Financial Economics*, 129, 479–509.

Bergstra J. and Bengio Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13, 281–305.

Blitz D.C. and van Vliet P. (2007). The volatility effect. *Journal of Portfolio Management*, 34, 102–113.

Connor G., Hagmann M. and Linton O. (2012). Efficient semiparametric estimation of the Fama-French model and extensions. *Econometrica*, 80, 713–754.

Coqueret G. and Guida T. (2018). Stock returns and the cross section of characteristics: A tree-based approach. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3169773.

DeMiguel V., Martin-Utrera A., Nogales F.J. and Uppal R. (2018). Transaction-cost perspective on the multitude of firm characteristics. Available at: https://ssrn.com/abstract=2912819.

Efroymson M.A. (1960). Multiple regression analysis. In Ralston A. and Wilf H.S. (eds.) *Mathematical Methods for Digital Computers*, John Wiley, New York.

Fama E.F. and French K.R. (1992). The cross-section of expected stock returns. *Journal of Finance*, 47, 427–465.

Fama E.F. and French K.R. (2015). A five-factor asset pricing model. *Journal of Financial Economics*, 116, 1–22.

Feng G., Polson N., and Xu, J. (2018). Deep learning factor alpha. Available at: https://ssrn.com/abstract=3243683.

Feng G., Giglio S. and Xiu D. (2019). Taming the factor zoo. Available at: https://www.ssrn.com/abstract=2934020.

Fischer T. and Krauss C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2), 654–669.

Frazzini A. and Pedersen L.H. (2014). Betting against beta. *Journal of Financial Economics*, 111, 1–25.

Frazzini A., Israel R. and Moskowitz T.J. (2018). Trading costs. Available at: https://ssrn.com/abstract=3229719.

Freyberger J., Neuhierl A. and Weber M. (2017). Dissecting characteristics nonparametrically. Available at: https://papers.ssrn.com/abstract=2820700.

Friedman J., Hastie T. and Tibshirani R. (2000). Additive logistic regression: A statistical view of boosting (with discussion). *Annals of Statistics*, 28, 337–407.

Friedman J., Hastie T. and Tibshirani R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1), 1–22.

Gedeon T.D. (1997). Data mining of inputs: Analysing magnitude and functional measures. *International Journal of Neural Systems*. 8(2), 209–218.

Goyal A. and Welch I. (2008). A comprehensive look at the empirical performance of equity premium prediction. *Review of Financial Studies*, 21, 1455–1508.

Green J., Hand R.M. and Zhang X.F. (2013a). The supraview of return predictive signals. *Review of Accounting Studies*, 18, 692–730.

Green J., Hand R.M. and Zhang X.F. (2013b). The remarkable multidimensionality in the cross section of expected US stock returns. *SSRN Electronic Journal*. Available at: https://www.researchgate.net/publication/256062440_The_Remarkable_Multidimensionality_in_the_Cross_Section_of_Expected_US_Stock_Returns.

Gu S., Kelly B. and Xiu D. (2018). Empirical asset pricing via machine learning. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3159577.

Guida T. and Coqueret G. (2018). Machine learning in systematic equity allocation: A model comparison. *Wilmott*, 98, 24–33.

Guida T. and Coqueret G. (2019). Ensemble learning applied to quant Equity: Gradient boosting in a multi-factor framework, in T. Guida (ed.). *Big Data and Machine Learning in Quantitative Investment*, Wiley, New York.

Harvey C.R., Liu Y. and Zhu H. (2016). … and the cross-section of expected returns. *Review of Financial Studies*, 29, 5–68.

Hastie T., Tibshirani R. and Friedman J. (2009). *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*, 2nd edition. Springer, New York.

Hinton G.E., Srivastava N., Krizhevsky A., Sutskever I. and Salakhutdinov R.R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. Available at: https://arxiv.org/abs/1207.0580.

Hornik K., Stinchcombe M. and White H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2, 359–366.

Horowitz J.L., Loughran T. and Savin N.E. (2000). Three analyses of the firm size premium. *Journal of Empirical Finance*, 7, 143–153.

Ioffe S. and Szegedy C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. Available at: https://arxiv.org/abs/1502.03167.

Krauss C., Do X.A. and Huck N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, 259(2), 689–702.

Li X., Chen S., Hu X. and Yang J. (2018). Understanding the disharmony between dropout and batch normalization by variance shift. Available at: https://arxiv.org/abs/1801.05134.

Macquarie Quantitative Research Report (2008). *Modelling the World*. Macquarie Quantitative Research. August 2008.

Macquarie Quantitative Research Report (2010). *Style Composites: Safety in Numbers*. Macquarie Quantitative Research.

Macquarie Quantitative Strategy Report (2019). *Self-driving Portfolios: The Artificial Intelligence Approach to Picking Stocks*. Macquarie Quantitative Research.

Messmer M. (2017). Deep learning and the cross-section of expected returns. Available at: https://ssrn.com/abstract=3081555.

Messmer M. and Audrino F. (2017). The (adaptive) LASSO in the zoo – Firm characteristic selection in the cross-section of expected returns. Available at: http://dx.doi.org/10.2139/ssrn.2930436.

Mezrich J.J. (1994). When is a tree a hedge? *Financial Analyst Journal*, 50, 75–81.

Moritz B. and Zimmermann T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. Available at: https://papers.ssrn.com/abstract=2740751.

Trippi R. and Efraim T. (1992). *Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real-world Performance*. McGraw-Hill, New York.

Zeiler M. (2012). ADADELTA: An adaptive learning rate method. Available at: https://arxiv.org/abs/1212.5701.

5

Enhancing Alpha Signals from Trade Ideas Data Using Supervised Learning

In this chapter, we present a methodology that uses supervised learning techniques, in particular "random forests" and "gradient boosting trees", to assess the probability of success on trade ideas. We use features relating to the underlying stocks' characteristics, the date of entry, along with features that describe the contributor of the idea. Additional information of the idea itself is also used. Our model improves the *ex post* performance of long ideas by more than 1%, on average, and the performance of short ideas by more than 2%.

5.1. Introduction

Institutional investors often use market experts' recommendations as an input in their investment decision process. Such recommendations vary significantly in accuracy. In this chapter, we present a method based on machine learning that assigns an estimated probability to a new idea, i.e. from 0 (least likely to be accurate) to 1 (most likely to be accurate). We then use these assessments to construct systematic investment portfolios. Forecaster information aside, this framework can be easily altered and applied in a universe of "theoretical" ideas that would select a "buy" or "sell" decision for each of a universe of stocks. There are two caveats to this projection. The first relates to the information content of the "forecaster" or "author" as a feature in the methodology. The second deals with the computational requirements in processing a much bigger universe of stocks and "theoretical ideas".

Chapter written by Georgios V. PAPAIOANNOU and Daniel GIAMOURIDIS.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Bailey *et al*. (2018) developed a ranking methodology to rank and grade market forecasters, in an attempt to understand the extent to which various forecasters have the forecasting skill. They observed four time horizons, namely one month, three months, nine months and two years, and weighted them using 0.25, 0.50, 0.75 and 1.0, respectively.

Accuracy across all forecasts in the dataset they used was 48%. They concluded that the forecasters' performance was random and that there was little skill involved, with a small number of them performing at accuracy levels 70–79% while the majority performing below 50%. They split the forecasters into investors and traders, showing slightly better performance, on average, within the traders group but still at 48.09%, i.e. below chance.

There is rich literature on the broader topic of analysts' forecasts, but the focus and the tools used have been different in most cases. Much of that literature is around analysts' earnings forecasts, and creating consensus forecasts and comparing with time series approaches. A good literature review is given by Bradshaw (2011). Although the objective and the approach of this chapter are different, the concept of reducing variance by forming an "ensemble" is central to our methodology that uses random forests. Furthermore, the analysis shows what metrics and quantities could have information content and are good candidates to be included as features for our machine learning approach.

For example, Jacob *et al*. (1999) and Clement (1999) argued that experience is a proxy for ability since it is an indicator that the analyst has not been fired due to poor performance. In our work, as will be explained later, we use the time difference from the first entry of the contributor to the database ("tenure"), as well as the number of ideas previously contributed ("engagement") and to-date hit rate ("performance"), as features in our machine learning algorithm. Bew *et al*. (2019) addressed the question of how to understand an analyst's forecasting ability by integrating track-record information from all stocks the analyst follows. They employ the independent Bayesian classifier combination (IBCC) method. The method was originally defined by Ghahramani and Kim (2003) and has been successfully applied in astronomy. It is ideally suited to sparse problems, enabling computationally efficient inference, dynamic tracking of analyst performance through time and real-time online forecasting. One of the limitations of their method is that there is no concept of ordering within the truth outcomes or the recommendations; they only get sets of categorical labels.

In this chapter, we use ensemble machine learning methods, specifically random forests and gradient boosting trees, to predict success or failure in trade ideas. These methods allow us to incorporate a reasonably wide range of predictors, with various levels of dependency on one another, as well as a way to look into their importance and predictive value. We evaluate the economic significance and practical implications of the approach we propose. We assign a probability to each trade idea, rebalance periodically by selecting from pooled trade ideas and retrain the model at a chosen frequency. We also span and explore a number of parameters, from model hyperparameters to the choice of training window, retraining frequency, among others. We find that momentum-related features are the most informative, as well as liquidity and volatility. We also find that improvements on long ideas range from 1.06% to 1.85%, whereas improvements on short ideas range from 1.57% to 2.78%. An even more striking result is that our approach has been successful in identifying short ideas with negative on average returns, in time periods when the equity markets were trending upwards.

The present work is organized as follows. Section 5.2 describes the main dataset along with the additional datasets it is augmented with, to come up with the predictive and predicted variables. Section 5.3 examines some key elements of the machine learning methodology and highlights the rationale behind some choices made. Section 5.4 presents the estimation results and feature importance of the machine learning method. In section 5.5, we attempt to explore the economic significance of the method by backtesting some of the many different strategies we can devise once a filtering mechanism on the ideas is available. The final section summarizes the key findings and results.

5.2. Data

We used a set of trade ideas that covered the period January 2005 to June 2019. The dataset contained approximately 790 thousand records and included equity trade ideas. Several fields were included, of which we used a subset that we found had good, consistent data quality. A subset corresponding to new ideas on American traded stocks was selected from the original dataset. We augmented this dataset with stock level information obtained through Axioma. The variables we considered fell into three broad categories: “idea”, “stock fundamental/technical” and “stock risk” related. In the first category, we considered an author’s engagement and tenure, among others; the second

category involved value, profitability and other information; the last category included idiosyncratic, total risk and others.

To provide a better understanding of the data we analyze, as well as motivate some key model development decisions we have made, we present in this section descriptive statistics of certain data attributes. Important features we highlight involve the time variation of ideas' success, patterns across long and short ideas and across the prediction horizon, the consistency of authors' success, and the database attrition. We conclude that classifying ideas *ex ante* involves modeling very complex dynamics that are more likely to be captured by machine learning techniques over linear methodologies.

Figure 5.1 shows the time variation of the raw ideas' success. The hit rate does not seem to exhibit a consistent trend of improving or deteriorating with time. Seasonality checks on a daily, weekly and monthly basis also do not show any obvious pattern to suggest the use of the day of the week or the month of the year as the predictive variable for the analysis.

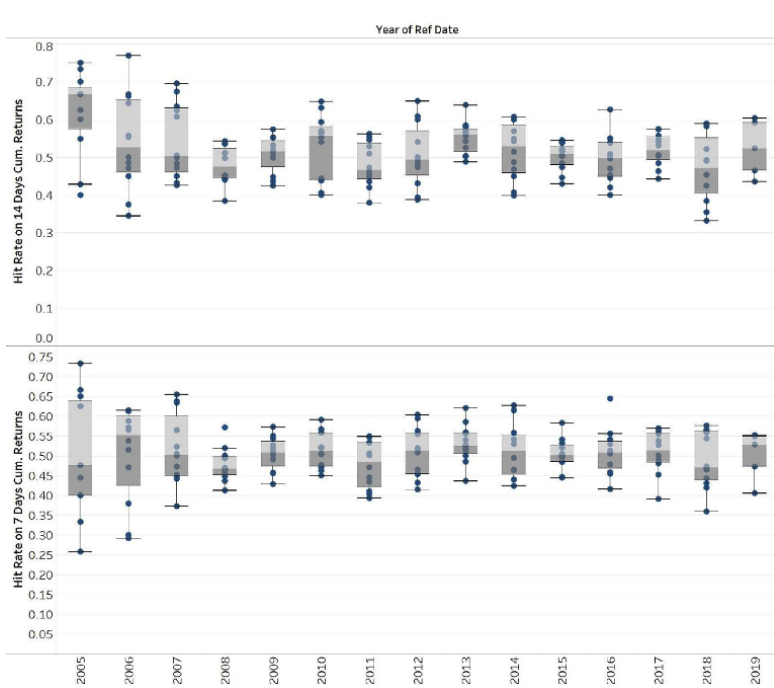


Figure 5.1. *Yearly variation of monthly "candlesticks" of the "hit rate" for ideas based on cumulative returns on 7- and 14-day horizons*

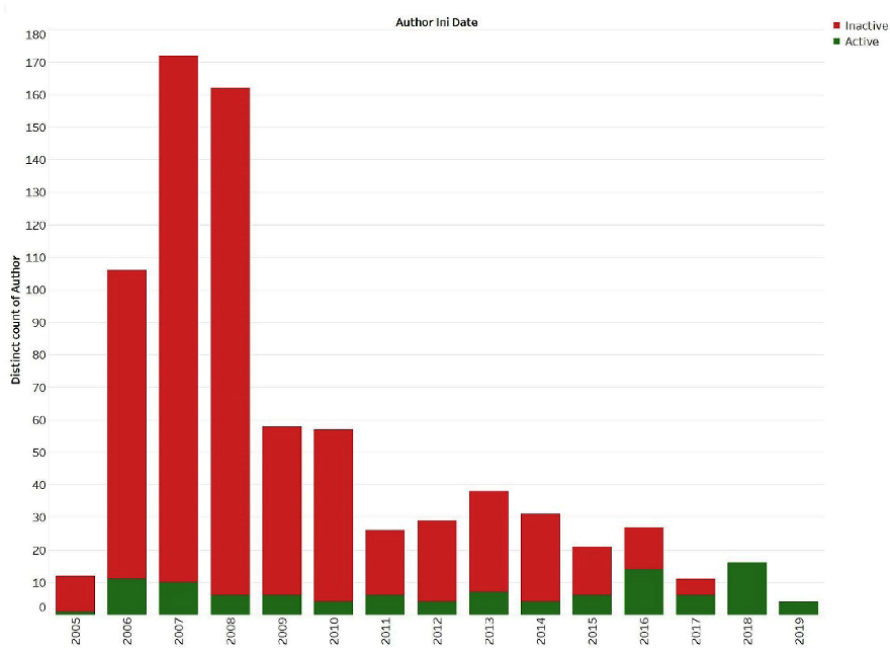


Figure 5.2. *Number of active and inactive authors by year of first appearance to the dataset. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

What insights do we get though when we turn to the contributor level information? Figure 5.3 shows the number of "effectively active" and "effectively inactive" authors by year of first appearance to the dataset. "Effectively active" is defined as an author that has contributed at least one idea after a chosen date, in this case January 1, 2018. Aggregating over the years, 105 out of 770 have contributed at least one idea after January 1, 2018. This suggests that the author name would not be a good feature to use for future inference. If we use the author name as a feature, the system would not be able to generalize to authors that would enter for the first time. This is the "cold start" problem, known in the application of recommender systems, a treatise of which can be found in Ricci *et al*. (2011). Furthermore, information from authors that have become inactive may not be reusable.

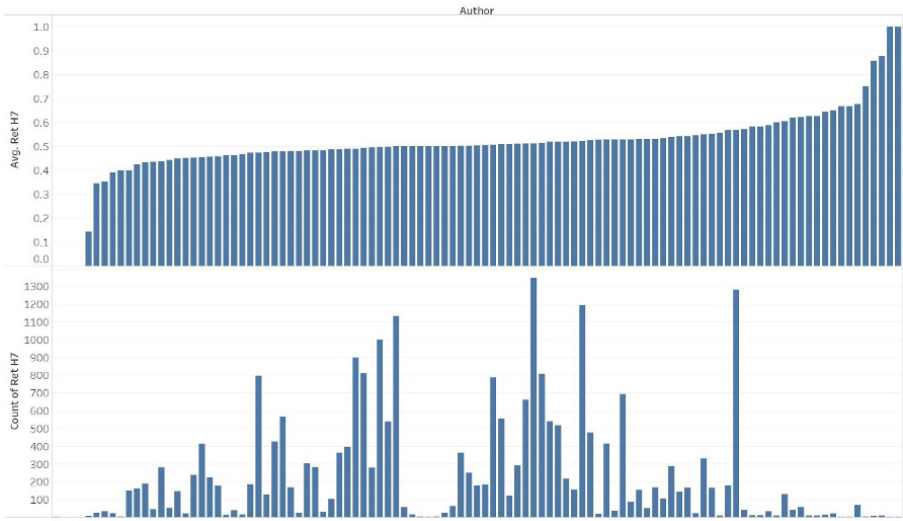


Figure 5.3. *"Active authors" hit rate at the seven-day cumulative returns horizon and corresponding number of ideas contributed by "active author"*

What complicates our objective even more is the persistence of accuracy in authors. Figure 5.3 shows the accuracy of each of the 105 effectively active users along with the number of ideas they have contributed. Twelve of those authors have a p value of less than 5% on the null hypothesis of expected average score of 0.5. The number of ideas for each of the 12 is: {2, 2, 2, 7, 7, 9, 69, 129, 151, 163, 284, 1326}. This means that from the 770 authors in the dataset, it is less than 12 that are still active and their accuracy bares statistical significance. Note that those numbers are the maximum number of ideas for each user in the whole dataset, i.e. by June 2019. It is important to note than when designing our features, we need to use only the number of ideas and performance up to the date corresponding to the record, to avoid the introduction of any forward-looking bias. Hence, based on the empirical observations from Figures 5.2 and 5.3 and the associated discussion, we obtain derived features such as the length of time the author has been in the database at the time they contributed the idea, among others.

Figure 5.4 shows the historic variation of hit rate on the eight-day horizon, along with the variation in the number of long and short direction ideas. It is evident that the number of long ideas contributed is significantly higher than the short ideas. The mean of long ideas across time is 0.5054, while that of short ideas across time is 0.4946, with a p value of 0.0226, which makes it somewhat ambiguous but tends to reject the null hypothesis, if a typical value of 5% threshold is selected, and indicates that there is significance in the higher prediction rate for the "longs". Perhaps this could be expected when considering that the number of longs is significantly higher and historically, there is an overall upward trend on the market returns.

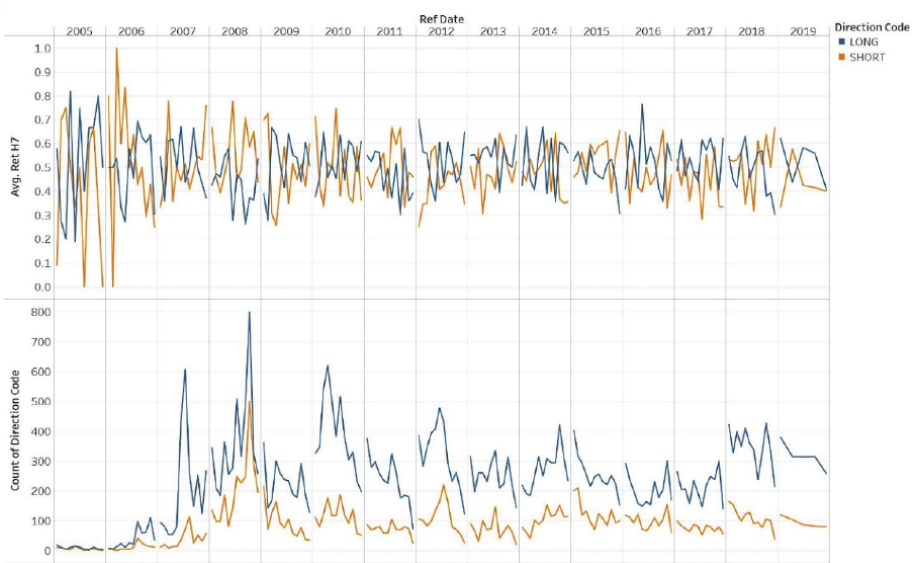


Figure 5.4. *Top: historical hit rates for long and short ideas by month. Bottom: number of long and short ideas historically by month. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

One final point in our empirical design is to determine the prediction horizon. Ideas reported in our database do not necessarily involve a time horizon. Anectodal evidence suggests that the typical prediction horizon ranges between 5 and 20 days. Figure 5.5 shows the hit rate based on

cumulative returns over different horizons from 1 to 30 days. There is a generally increasing trend on the hit rate, with the very short term being lower. This is consistent with observations by Bailey *et al*. (2018), among others, who explained this in terms of more noise on the short-term horizon.

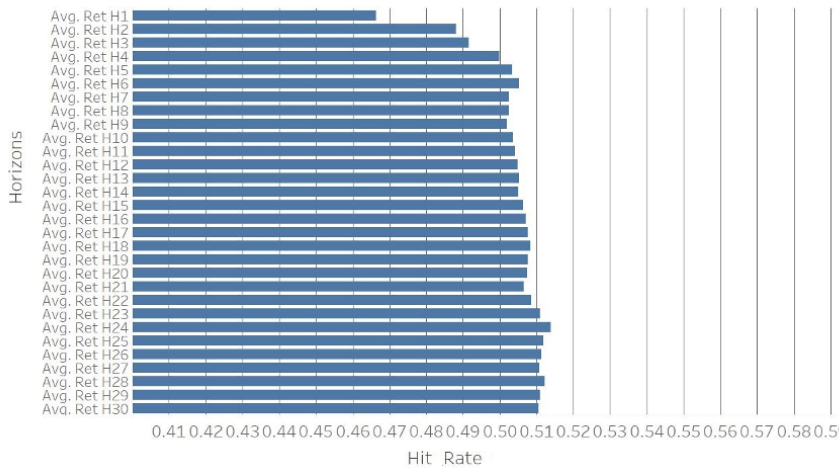


Figure 5.5. *Hit rate based on cumulative returns over different horizons from 1 to 30 days*

5.3. Model and empirical design

In this section, we discuss the model we use, i.e. the statistical framework and the input variables, as well as the design of our estimation and backtesting frameworks.

We employ a supervised learning approach using binary classification, with the two classes representing a view on whether an idea is "likely" or "unlikely" to be successful. Many of the classification methodologies actually compare a probability or functional output to a threshold value, often 0.5, before they convert the prediction to a binary class. Logistic

regression, artificial neural networks (ANNs) and ensemble methods, such as random forests, all provide such information. We focus on two methods, namely random forests and gradient boosting trees.

The labeling for the training set is defined as follows:

$$ClassLabel_i = sign(Direction_i) * sign(CumReturns_i)$$

where

$$sign(Direction_i) = \begin{cases} 1, if\ Direction\ is\ Long \\ 0, if\ Direction\ is\ Short \end{cases}$$

In the training phase, we fit a model by minimizing the log-loss objective function.

Note that when the model is used for inference, the time horizon relevant to our prediction should be consistent with the time horizon the model has been trained on. Considering that there is some variability in the hit rates based on the horizon, as shown in Figure 5.5, a different model is trained for each time horizon. The relevant horizon range is determined by the rebalance frequency. The idea, long or short, has to generate positive returns by the rebalance period when the new ideas are pooled and considered. Different variations and extensions can be constructed on how we bring in the new ideas and what we do with those existing; however, in this work, for simplicity, we pool the ideas between rebalance periods, construct new portfolios based on the predictions on those and scrap the old ideas from the previous rebalance.

Random forests: this is a well-established and well-documented method, with rich literature following the early works by Ho (1995). The method creates ensembles of hundreds or thousands of individual decision trees. Central to the method are the concept of "bagging" and "bootstrapping". The latter refers to the resampling of the original dataset, with replacement, a rather old concept known from the early works of Efron (1979) and before. Bagging, or bootstrap aggregating, includes an aggregation step to form majority vote and reach a final prediction.

Combining predictions from multiple models in ensembles works better if the predictions from the sub-models are uncorrelated or, at best, weakly correlated. Decision trees choose which variable to split on, using a greedy algorithm that minimizes error. As such, even with bagging, decision trees can have many structural similarities and in turn have high correlation in their predictions. Random forest changes the algorithm for the way that the sub-trees are learned so that the resulting predictions from all of the sub-trees have less correlation. In a classification and regression tree (CART) method, when selecting a split point, the learning algorithm is allowed to look through all variables and all variable values, in order to select the most optimal split point.

The random forest algorithm changes this procedure so that the learning algorithm is limited to a random sample of features for which to search. The number of features that can be searched for at each split point must be specified as a parameter to the algorithm. Each of the trees is allowed to use a random selection of M variables. As a rule of thumb for classification problems, a good default value is:

$$M = \sqrt{(Number\ of\ Features)}$$

In our case, where we have about 25 variables, this value is 5; however, in any case, it is a parameter that we can perform our hyperparameter tuning on, along with the number of trees. Ensembling many resampled decision trees serves to reduce their variance, producing more stable estimators that generalize well out-of-sample. Random forests are hard to overfit and generalize well without extensive need for tuning. They also have an additional computational advantage: they are embarrassingly parallel and therefore scale much better to larger datasets.

Gradient boosting machines (GBM): introduced by Friedman (2001), is a generalization of boosting to arbitrary differentiable loss functions. They build on the boosting ideas of AdaBoost, introduced by Freund and Schapire (1999). Gradient boosting regression trees (GBRT) offer an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems. Some of the advantages of GBRT include the handling of mixed-type, heterogeneous features and their robustness to

outliers. The disadvantage is scalability; due to the sequential nature of boosting, it cannot be parallelized across trees in the same way as random forests can. To clarify the idea of "boosting", the trees are produced in sequence to assign extra weight to the previously misclassified data points. This means that one tree has to be completed before the next, as it requires information from the previous tree, i.e. the weighting of the data points in the loss function. An efficient and scalable implementation is offered in the XGBoost (eXtreme Gradient Boosting) package.

While the statistical framework is very important, equally important in a supervised learning framework are the input variables, i.e. the "regressors". We use a total of 25 features (see Table 5.1), both numerical and categorical. The features in the fundamental/technical category are either the exposure of the stocks to corresponding fundamental factors or market data. For example, "value", "leverage", "profitability", "growth", "earnings yield", "size", "market sensitivity", "medium-term momentum", "exchange rate sensitivity", "liquidity", "volatility" and "industry" belong to the first category. "Average daily volume" is market data available from the risk model provider. Total risk, specific risk, predicted beta and specific returns are reported by the third-party risk model. We report the number of missing values, as computed post a first pass of elimination of ideas that could not be mapped in all datasets, i.e. very "sparse" horizontally. Univariate "imputing" was used for the remaining missing values. For numerical features, we use a "mean" value. For categorical variables, we use one of the two following methods of imputing missing values: "constant" where the missing values are replaced with a provided "fill_value" such as zero, and "most frequent" where the missing values are replaced with the most frequent value along each column, but as we see in this case, only numerical features have missing values. For the categorical data, we used "ordinal" encoding as, among the various choices, it generally works equally well or better for tree methods, and requires fewer computational resources.

We pursue proper out-of-time validation (OTV) to avoid the issues of overfitting and a favorable "forward-looking" bias. Figure 5.6 shows our approach for OTV. To tune the hyperparameters of the models, we use rolling trailing and development/testing windows, with the testing window always succeeding the training window. A holdout corresponds to the true testing period where the frozen models are tested on data not seen while training and tuning.

Feature type	Feature name	Var type	Unique	Missing
Idea	DIRECTION_CODE	Categorical	2	0
Idea	Author tenure	Numerical	3998	0
Idea	Author engagement	Numerical	1339	0
Idea	Author performance	Numerical	17487	0
Idea	Author classification	Categorical	3	0
Idea	|Target price–security price|	Numerical	22589	2832
Fundamental/technical	Value	Numerical	31063	0
Fundamental/technical	Leverage	Numerical	29987	0
Fundamental/technical	Profitability	Numerical	31047	0
Fundamental/technical	Growth	Numerical	30971	0
Fundamental/technical	Earnings yield	Numerical	30913	0
Fundamental/technical	Size	Numerical	31036	0
Fundamental/technical	Market sensitivity	Numerical	31081	0
Fundamental/technical	Medium-term momentum	Numerical	31036	0
Fundamental/technical	Liquidity	Numerical	25505	0
Fundamental/technical	Volatility	Numerical	31087	0
Fundamental/technical	Exchange rate sensitivity	Numerical	31238	23
Fundamental/technical	Industry	Categorical	68	0
Fundamental/technical	Average daily volume	Numerical	31251	13
Fundamental/technical	Specific return	Numerical	30805	352
Fundamental/technical	Last move	Categorical	2	0
Risk	Total risk	Numerical	31109	0
Risk	Specific risk	Numerical	31366	0
Risk	Predicted beta	Numerical	9994	0

Table 5.1. *Key features used*

Figure 5.6. *Illustration of out-of-time validation (OTV). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

For the period of the backtest and for the given training frequency, a number of models are trained for each horizon. For example, assume we run a backtest spanning a year, where we choose to retrain the machine learning model every three months. The training is done on a trailing window of six years from the day of training. In the course of a years' backtest the models will retrain four times. The number of models to retrain is equal to the number of relevant horizons. Then, as we iterate through the rebalance dates – say weekly – we use the set of models that has end of training time as close to – but not exceeding, the rebalance date. This set has the same training window but different labeling, based on the returns horizon used. For the weekly rebalance, the relevant horizon is from 8- to 14-day returns in order to take into account the aging of the idea. However, an idea that has been pooling for a few days by the time of the next rebalance can be assessed on its success while pooled, and extra information can be incorporated.

5.4. Estimation and robustness

In this section, we present key results of the estimation of machine learning models, as well as an analysis of the robustness of the models against key choices we made.

First, we discuss the hierarchy of the features. We pursue a technique called permutation importance. This technique involves altering input data and observing the effect on the model score, i.e. to measure how much worse a model's error score could be if predictions were made after randomly shuffling that column, while leaving all other columns unchanged. We

present the results of this analysis in Figure 5.7. The DIRECTION_CODE is deemed the most important feature. This is the categorical feature that describes whether an idea is long or short. This is somewhat intuitive when considering the fact that a flip in the DIRECTION_CODE would flip the label, i.e. whether the idea is successful or not. The second and third most important features are "specific risk" and "specific return". The specific return is essentially the stocks' alpha capacity, which is, in a way, what our approach aims to capture in the first place. Collectively, the hierarchy shows stark consistency with the conclusion reached by Gu *et al*. (2019) that there are ultimately three types of variables that are important: momentum, liquidity, volatility. The strict order, as shown in Figure 5.7, is not fully robust to changes in model parameters. Flips in the relative importance can be observed between features, but overall the groups that appear on the top and bottom positions are generally consistent.

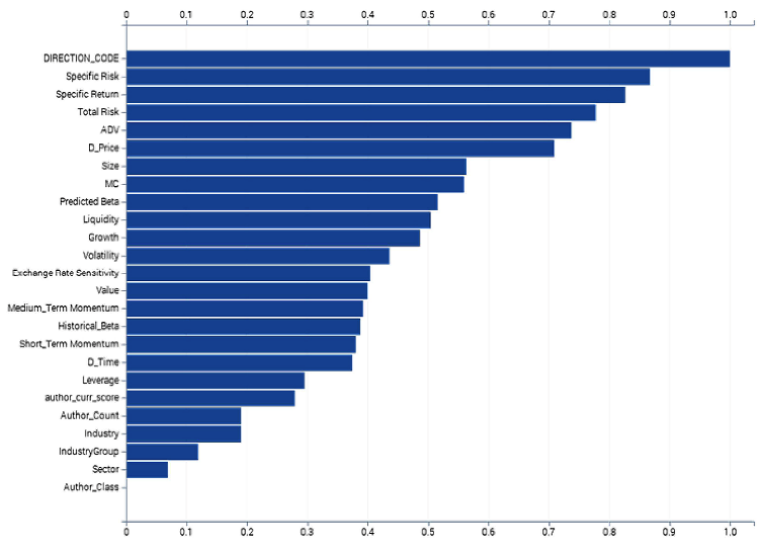


Figure 5.7. *Feature importance normalized to the most important feature*

We next examine the direction of the features' importance. We wish to establish whether, for example, "specific return" is positively associated with an idea's success. We present "partial dependence" plots to shed light on this question for select variables, namely "specific return" and "tenure". Figure 5.8 shows that we are more likely to correctly predict the stocks with higher specific returns, i.e. there is a positive correlation between idea success and observed specific return. A similar plot in Figure 5.9 shows the partial dependence on the

number of days elapsed between the idea date and the first time the same author appeared in the database, or "tenure". There is an interesting jump on 820–860 days. Then, it seems that the performance deteriorates slowly with increasing tenure. When examining similar graphs for other features, we reach the general conclusion that in some, those dependencies are not monotonic and argue that they would likely not be captured in a multi-linear regression setting where the betas would represent the slope in a liner fit.

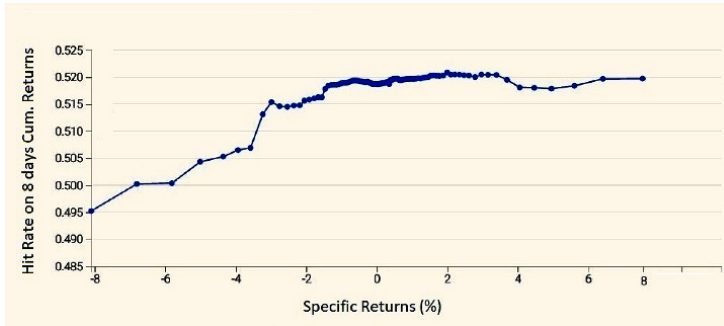


Figure 5.8. *Partial dependence of hit rate on eight-day cumulative returns on specific returns*

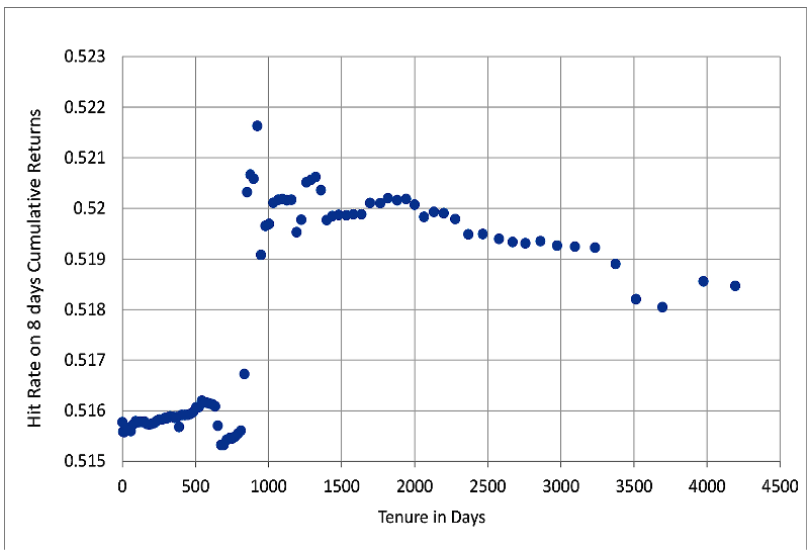


Figure 5.9. *Partial dependence of hit rate on eight-day cumulative returns on author "tenure"*

We now turn to the robustness of the model against the choice of the model's training time. In this analysis, we tested different training windows to assess the best span of training time. There are a few considerations in play. Apparently, the longer the training window, the more computationally expensive the model becomes and the more it becomes prone to potential structural breaks. Figure 5.10 shows the values of the log-loss function, i.e. our objective function, over 12-month test sets for different training periods in months. It should be seen in the context of Figure 5.6[1]. Each training duration would change the length of the dashed boxes. We find little variation in the minimum error with increasing training window. In fact, besides the first 64-month period, the "holdout" error is somewhat higher; the variations thereafter are very small and could be due to hyperparameter tuning, with one case potentially being better tuned than the other.

Figure 5.10. *Log-loss function for the eight-day returns horizon label versus training period in months. Blue corresponds to the testing period of 12 months, July 2017–June 2018. Orange corresponds to the average of 12-month rolling windows. Gray is the period from July 2018 to June 2019. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

1 The first bar (blue) in Figure 10 corresponds to the log-loss function of the second top bar shown in Figure 6. The second bar (orange) corresponds to the ensemble of several one-year windows, i.e. the green part of all the bars of Figure 6, except the top. The last family of bars in Figure 10 (gray), corresponds to the "hold-out" on the top bar of Figure 6.

We next assess the model's fit using the area under the receiver operating characteristic (ROC) curve. This method has the benefit of being invariant to the threshold used to attribute a prediction to one class or the other. A default value for threshold is 0.5, but any value deemed appropriate for the problem could be defined. Figure 5.11 shows an ROC curve derived for a gradient boosting tree classifier backtest period of one year, June 2017–June 2018, and training period, March 2009–June 2017, for the eight-day cumulative returns horizon. A diagonal ROC resulting in AUC of 0.5 is essentially a random classifier.

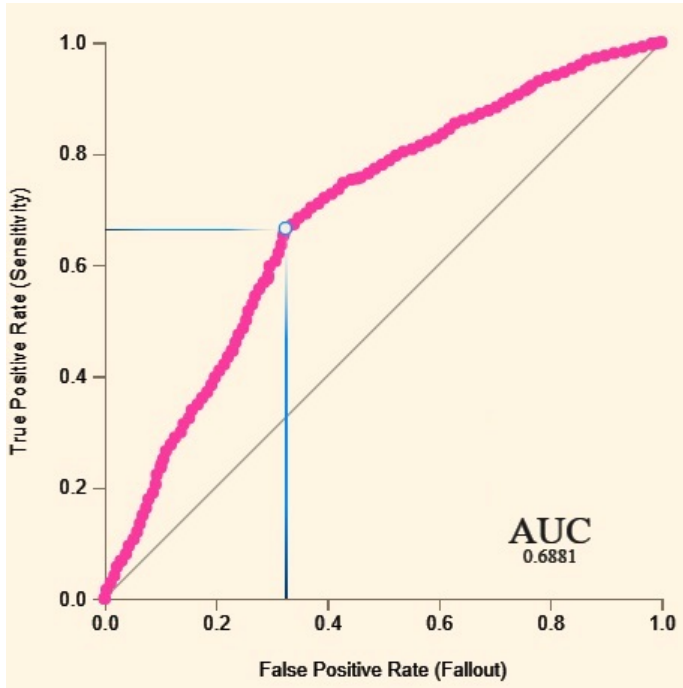


Figure 5.11. *Receiver operating characteristic (ROC) curve for the gradient boosting tree classifier backtest period of one year, June 2017–June 2018, and the training period from March 2009 to June 2017, for the eight-day cumulative returns horizon. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

As mentioned, a selection based on performance is made between different variants of gradient boosting tree and random forest methods. Within each family of methods, some hyperparameter tuning on a fairly large number of parameters is possible. Figure 5.12 shows sensitivity to the number of trees

and the maximum depth of trees allowed for the gradient boosting trees method. Other parameters tuned include the learning rate and the maximum number of features allowed to be selected per split node. There are several other hyperparameters that we did not attempt tuning on, as the dimensionality would become very large and we could end up overfitting. Moreover, as shown in the figures, there is some effect on the hyperparameter tuning, but it only affects the third decimal on the loss function, and up to 0.03 on the AUC metric. However, note that there is an interplay between the various parameters and a grid search or randomized search is often necessary in a multidimensional grid of tunable parameters. For example, decreasing the learning rate causes an increase in the optimal number of iterators.

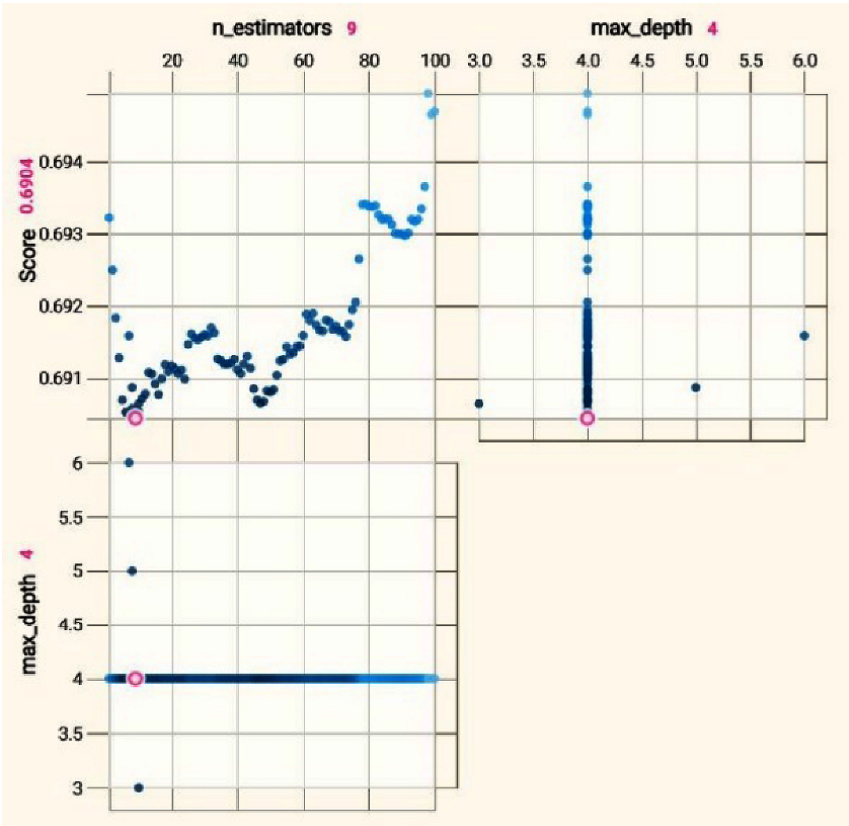


Figure 5.12. *Hyperparameter tuning example for the gradient boosting tree, showing tuning on the number of trees and the maximum depth of trees. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Figure 5.13 shows the example of hyperparameter tuning on random forests' maximum tree depth and the maximum number of variables that can be searched in each node split. Note that generally a good default value for the latter is the square root of the number of features, which would be 5, since our total number of features is 25; however, the hyperparameter tuning seems to have selected the maximum number of the range provided, which is 12.

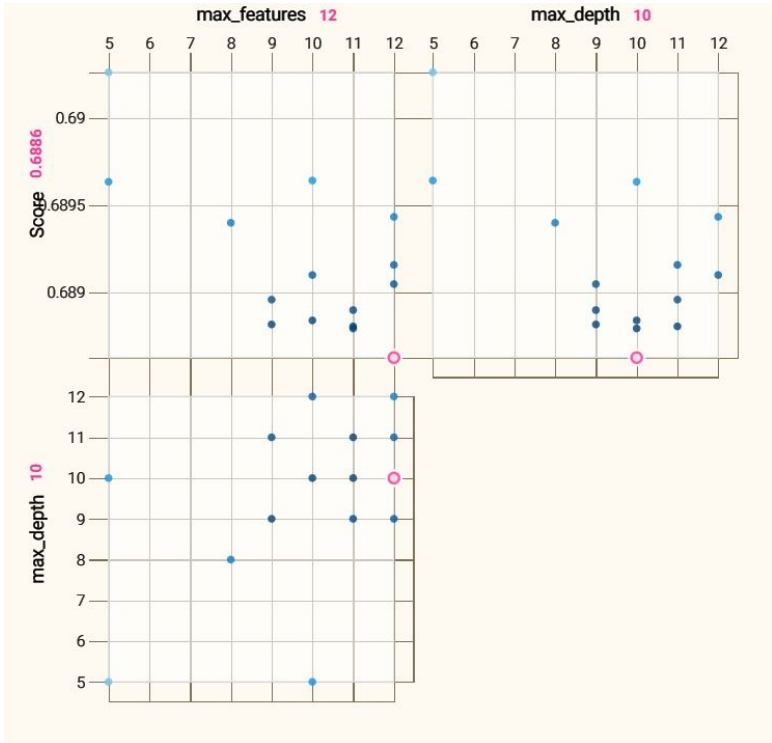


Figure 5.13. *Hyperparameter tuning example for random forest, showing tuning on the number of features that can be searched in each split and the maximum tree depth. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Note, that by increasing the maximum tree depth, each tree is capturing more detail, which can lead to higher variance. This additional variance is then compensated by further increasing the number of iterators, or trees. Similarly, the smaller the number of features allowed to be randomly sampled on each node, the lower the variance.

In summary, this section investigated the ability of the machine learning models to study the problem at hand and the robustness against key choices we made. We concluded that momentum, liquidity and volatility stand out in the hierarchy of important features. We further verified that our training period is well chosen and robust. Moreover, we showed that the model hyper-parameters are satisfactorily tuned.

5.5. Economic significance

In this section, we discuss the economic significance of the predicted ideas. Implementing a systematic investment strategy, based on the predictions our model produces, involves numerous additional implementation choices and is beyond the scope of this work. We thus evaluate the economic significance of our predictions based on the potential improvement they offer over the ideas in their original form.

We present in Figure 5.14 the distribution of cumulative weekly unsigned returns for each idea gathered during 2016–2018. Long and short ideas are collected and displayed separately. We observe that in all cases, the filtering results in a shift in the "long" distribution towards the right, i.e. towards higher returns, while for the "short" distribution, there is a shift towards the left, i.e. towards negative returns, which is the desirable direction since those names are suggested to be shorted.

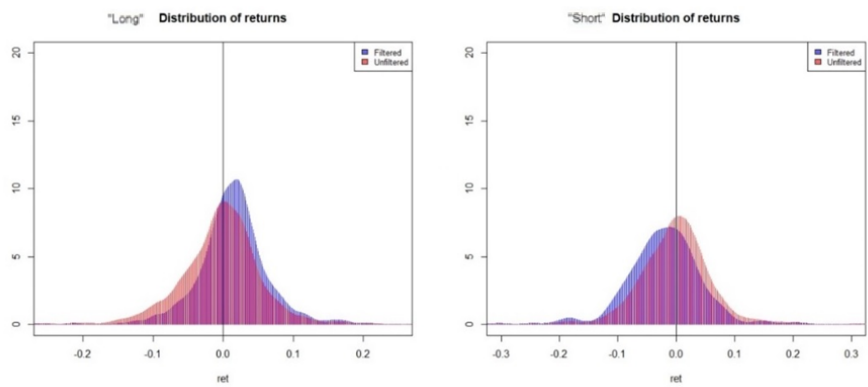


Figure 5.14a. *Distribution of cumulative weekly returns for the 2018 backtest. Comparison between filtered and unfiltered ideas for long and short ideas separately. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

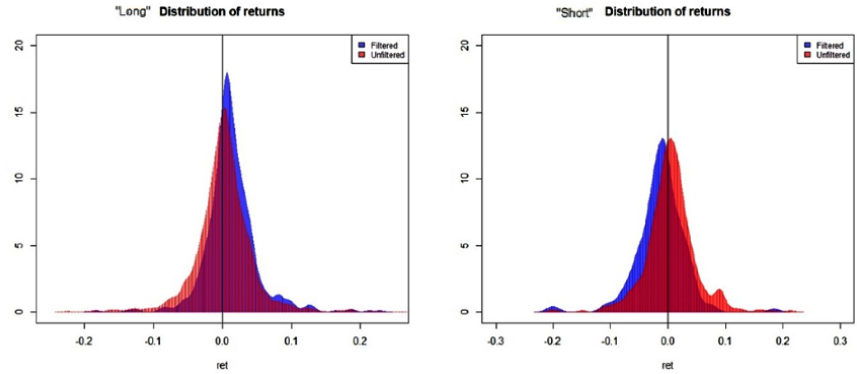


Figure 5.14b. *Distribution of cumulative weekly returns for the 2017 backtest. Comparison between filtered and original set of ideas for long and short ideas separately. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

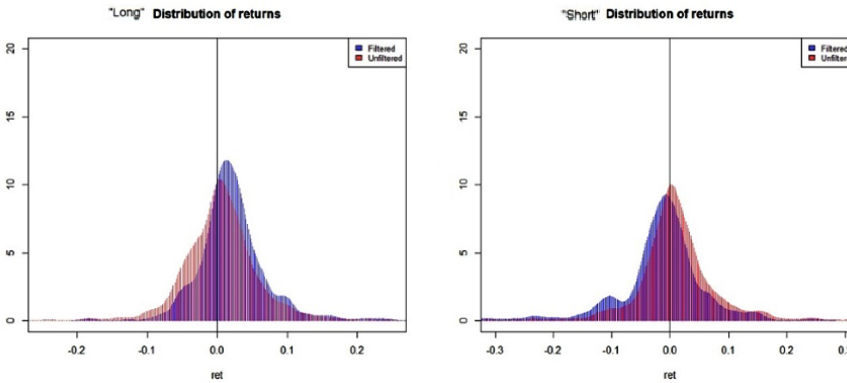


Figure 5.14c. *Distribution of cumulative weekly returns for the 2016 backtest. Comparison between filtered and original set of ideas for long and short ideas separately. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

We also present in Table 5.2 the mean cumulative weekly returns for 2016–2018 and half-yearly and quarterly frequencies. A window of five years trailing the retraining date is used. The results confirm the shift of the mean towards higher values for long ideas and lower values for short ideas. In particular, we find that improvements on long ideas range from 1.06% to 1.85%, whereas improvements on short ideas range from 1.57% to 2.78%.

An even more interesting result, in our view, is that our approach has been successful in identifying short ideas with negative on average returns, in time periods when the equity markets were trending upwards.

		Long			Short		
Backtest year	**Training frequency (days)**	**Mean returns ML**	**Mean returns original**	**Delta mean returns**	**Mean returns ML**	**Mean returns original**	**Delta mean returns**
2016	92	0.0251	0.0066	**0.0185****	-0.0169	0.0108	**-0.0278****
2016	183	0.0244	0.0066	**0.0177****	-0.0161	0.0108	**-0.0269****
2017	92	0.0127	0.0021	**0.0106****	-0.0130	0.0045	**-0.0175****
2017	183	0.0132	0.0021	**0.0111****	-0.0112	0.0045	**-0.0157****
2018	92	0.0144	-0.0032	**0.0175****	-0.0200	0.0014	**-0.0214****
2018	183	0.0140	-0.0032	**0.0172****	-0.0194	0.0014	**-0.0208****

Table 5.2. *Mean cumulative weekly returns for long and short, filtered and unfiltered ideas, per backtest year and for different training frequencies using a five-year training window. **Significance at the 1% significance level*

5.6. Conclusion

This chapter applies machine learning in the context of filtering trade investment ideas. More specifically, we use random forests and gradient boosting trees in order to predict success or failure in trade ideas. These methods allow us to incorporate a reasonably wide range of predictors, with various levels of dependency on one another, as well as a way to look into their importance and predictive value. We evaluate the economic significance and practical implications of the proposed approach. We assign a probability to each trade idea, rebalance periodically by selecting from pooled trade ideas and retrain the model at a chosen frequency. We also span and explore up on a number of parameters, from model hyperparameters to the choice of training window, retraining frequency, among others. We find that momentum-related features are the most informative, as well as liquidity and volatility. We also find that improvements on long ideas range from 1.06% to 1.85%, whereas improvements on short ideas range from 1.57% to 2.78%. An even more striking result is that our approach has been successful in identifying short ideas with negative on average returns, in a period of upward-trending overall equity market.

5.7. References

Ahn H.J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178 (1), 37–51.

Bailey D.H., Borwein J.M., Salehipour A. and de Prado, M.L. (2018). Evaluation and Ranking of Market Forecasters. *Journal of Investment Management*, 16 (2), 47–64

Bew D., Harvey C.R., Ledford A., Radnor S. and Sinclair A. (2019). Modeling Analysts' Recommendations via Bayesian Machine Learning. *The Journal of Financial Data Science*, 1, 75–98.

Clement M. (1999). Analyst Forecast accuracy: do ability, resources, and portfolio complexity matter? *Journal of Accounting and Economics*, 27, 285–303

Efron B. (1979). Bootstrap methods: Another look at the jackknife. *Annals of Statistics*, 7 (1), 1–26.

Freund Y. and Schapire R.E. (1999). A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14 (5), 771–780.

Friedman J.H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29, 1180.

Ghahramani Z. and Kim H-C. (2003). Beyesian Classifier Combination, Gatsby Computational Neuroscience Unit Technical Report, No. GCNU-T. London, UK.

Gu S., Kelly B.T. and Xiu D. (2019). Empirical Asset Pricing via Machine Learning. Chicago Booth Research Paper No. 18-04. *31st Australasian Finance and Banking Conference 2018*. Available at: http://dx.doi.org/10.2139/ssrn.3159577

Ho T.K. (1995). Random decision forests. *Proceedings of the Third International Conference on Document Analysis and Recognition*. IEEE.

Jacob J., Lys T.J. and Neale M.A. (1999). Expertise in Forecasting Performance of Security Analysts. *Journal of Accounting and Economics*, 28 (1), 51–82.

Mark T. Bradshaw. (2011). Analysts' Forecasts: What Do We Know After Decades of Work? June 30. Boston College.

Papernbrock J. (2011). Asset Clusters and Asset Networks in Financial Risk Management and Portfolio Optimization. PhD thesis. KIT, Karlsruhe.

Ricci F., Rokach L., Shapira B. and Kantor P.B. (2011). *Recommender Systems Handbook*, Springer, New York.

6

Natural Language Process and Machine Learning in Global Stock Selection

6.1. Introduction

Today's investing world is characterized by information overload. Traditionally, investors focus on numerical data, using either fundamental analysis or quantitative models. However, the vast majority of the available information is in unstructured formats, for example text, audio, video and image. Given that most analytical techniques are designed to analyze numerical data rather than unstructured information, the first challenge is how to make such intelligence machine readable. Natural language processing (NLP) is a great example – it quantifies and transforms textual data into tangible information. Analyzing textual information, however, goes far beyond NLP. The process requires a suite of integrated technology and analytical systems, across various facets spanning web scraping, pruning, distributed parallel computing, NLP algorithms, topic modeling and machine learning.

6.1.1. *The performance of traditional stock selection factors continues to shrink*

The average return of common stock selection factors and models has compressed considerably over the past 30 years. Although fundamental and discretionary managers may not use factors explicitly, their investment process often involves buying stocks that are cheap, good quality and highly rated by

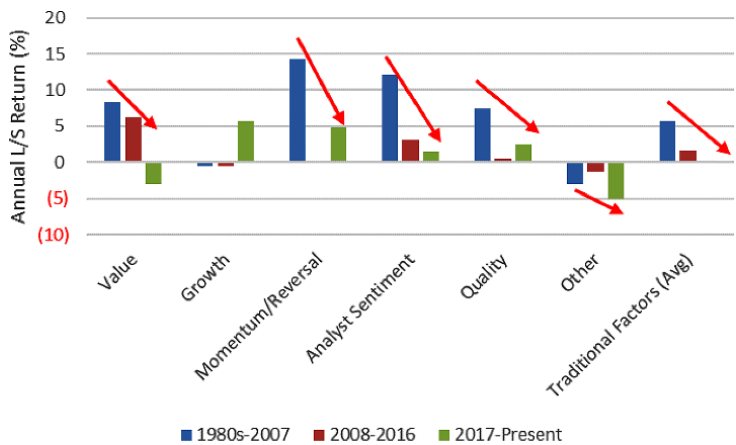
Chapter written by Yin LUO.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

sell-side analysts. Therefore, fundamental portfolios are often correlated to value, profitability and momentum factors.

As shown in Figure 6.1a, the average alpha associated with most investment styles[1] has plunged in the past 30 years, especially for value, momentum, analyst sentiment and quality – arguably the most well accepted and understood anomalies by both academics and practitioners. The average excess return of our 15 common factors since 2017 is essentially zero. Growth factors, which were traditionally rather weak signals, are the only exception and have been able to preserve some of their performance in the post-2017 environment.

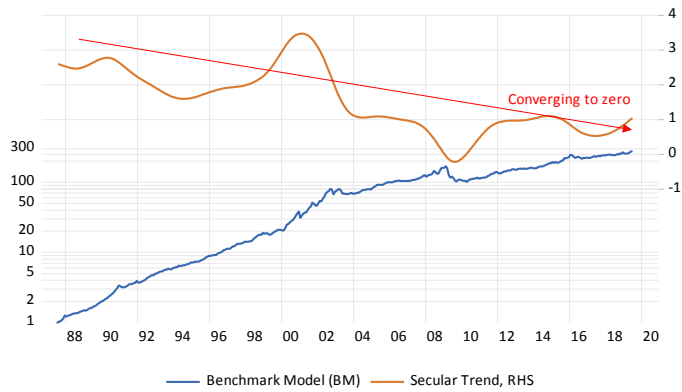
Given the declining pattern of factor performance, it is not surprising to see that traditional multifactor models have also struggled (Figure 6.1b). The alpha from plain vanilla strategies such as the BM[2] (Benchmark Model) is approaching zero, albeit with some cyclical ups and downs.



a) Average long/short quintile portfolio return, US

1 In this section, we use 15 common stock selection factors (also known as smart beta or risk premia factor portfolios) from six styles, as examples. In particular, we use earnings yield, dividend yield, book-to-market and EBITDA/EV to represent value; consensus five-year expected growth and FY1/FY0 expected earnings growth for growth style; 12-month return price momentum and one-month reversal for momentum/reversal; FY1 earnings revision for analyst sentiment; ROE (Return on Equity), debt/equity ratio and Sloan (1996) accruals for quality; and low beta, illiquidity and size (small cap) for other styles.

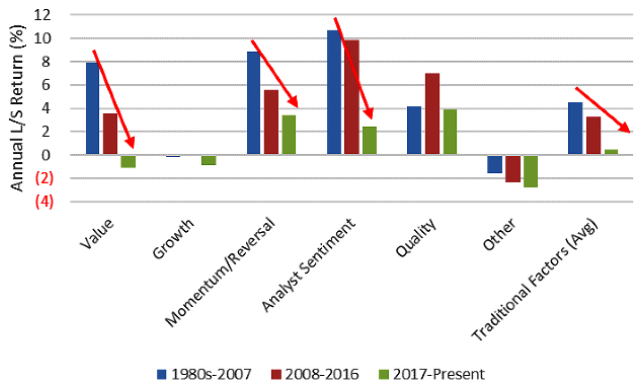
2 The BM weighs the following eight factors equally: earnings yield, book-to-market, FY1/FY0 expected earnings growth, FY1 earnings revision, ROE, debt/equity ratio and Sloan (1996) accruals.



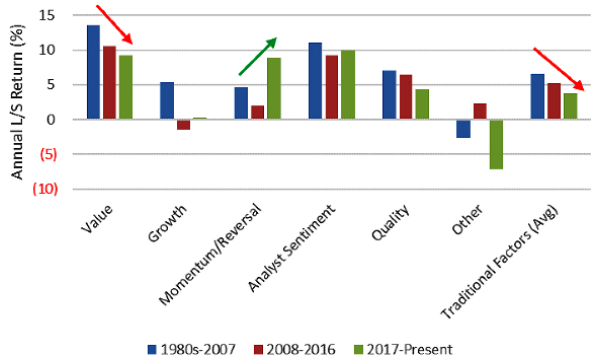
b) Cumulative performance of the BM

Figure 6.1. *Traditional factor performance continues to struggle in the US. (Sources: Bloomberg Finance LLP, EDGAR, FTSE Russell, Haver, I/B/E/S, S&P Capital IQ, Thomson Reuters, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

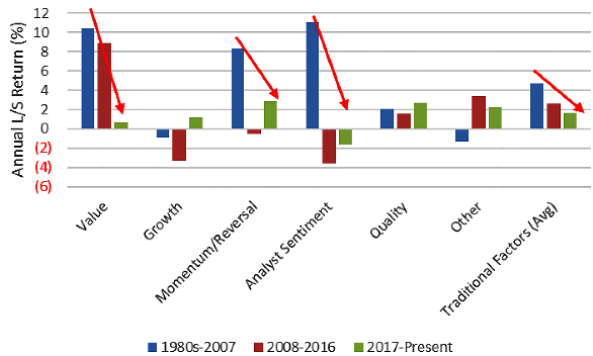
Outside of the US equity market, we observe similar patterns around the world. In Europe, value, momentum and analyst sentiment factors suffered the largest defeat in recent years (Figure 6.2a). While factor performance in Asia ex-Japan has held up materially better, value and growth signals have deteriorated considerably (Figure 6.2b). Japan is not an exception either. As shown in Figure 6.2c, most factors and styles also plummeted over time. Lastly, as shown in Figure 6.2d, typical multifactor models have seen their performance flattened (in Europe and Asia) or retreated (in Japan).



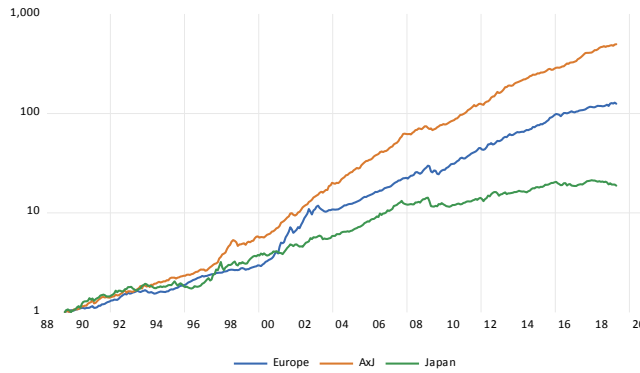
a) Europe



b) Asia ex-Japan



c) Japan



d) BM in Europe, Asia and Japan

Figure 6.2. *Traditional factor performance, global. (Sources: Bloomberg Finance LLP, EDGAR, FTSE Russell, Haver, I/B/E/S, S&P Capital IQ, Thomson Reuters, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

As discussed in Luo *et al*. (2019), there are both cyclical reasons (e.g. shift in macroeconomic environment, international trade uncertainty, ultra-loose monetary policies by major central banks) and long-term secular trends (e.g. active investors are increasingly more efficient in exploring market anomalies) behind the decline in factor performance.

As the performance from traditional stock selection factors continues to shrink, alternative data and machine learning are likely to add uncorrelated alpha. In this chapter, we focus on using NLP (natural language processing) techniques to generate insights from textual data. Furthermore, we illustrate how behavioral finance and machine learning techniques can be used to improve traditional news sentiment signals.

6.1.2. *Textual data, natural language processing and machine learning*

Traditionally, investors focused on numerical data, using either fundamental analysis or quantitative models. However, the vast majority of the available information is in textual format. For example, there are almost 5,000 documents filed by public companies in the US every day. On average, there are over 150,000 words in a typical 10-K – the standard annual filing of a company's performance required by the SEC (Securities and Exchange Commission). Companies communicate their performance and strategies to the public via investors' presentations and conference calls. Investors do not have enough time to sift through this mountain of written material. Furthermore, human interpretation is subjective and certainly exposed to behavioral biases.

Das and Chen (2001) and Antweiler and Frank (2002) are among the first researchers to study textual information through NLP. The authors used linguistic methods to examine the effect of messages posted on *Yahoo! Finance* and *Raging Bull*, for the companies in the Dow Jones Index. Other well-known early works include Li (2006) and Tetlock *et al*. (2008). Li (2006) analyzes sentiment from corporate filings and finds that certain words in firms' annual reports predict low annual earnings and stock returns. Tetlock *et al*. (2008) quantifies sentiment based on news stories rather than annual reports. He suggests that the fraction of negative words in firm-specific news stories leads to low subsequent earnings. For the tone analysis, the study of Tetlock *et al*. (2008) is based on a general-purpose

Harvard dictionary, which is then further refined by Loughran and McDonald (2011). They suggest that the Harvard psychological dictionary may not be suitable for finance and accounting applications because the meaning of positive and negative words may have very different connotations in a financial context. Loughran and McDonald (2011) develop a finance-specific sentiment dictionary. They are also among the first to extensively apply NLP on the SEC Form 10-K filings and link the signals to stock returns, trading volume, return volatility, fraud, material weakness and unexpected earnings. Rohal *et al*. (2017) develop a comprehensive suite of algorithms and stock selection signals, based on SEC's EDGAR filing system. Similarly, Wang *et al*. (2017b) find that insights from EDGAR filings can be useful in predicting takeover targets.

On company call transcripts, Ahmad and Zinzalian (2010) is one of the original studies. Ahmad and Zinzalian find that, with large enough training data, historical volatility together with techniques such as POS tagging[3] can improve stock volatility forecasting over pure statistical models. Price *et al*. (2011) propose that conference call linguistic tone is a significant predictor of abnormal return and trading volume. The Q&A section of the call has incremental explanatory power for the Post-Earnings-Announcement-Drift (PEAD)[4]. Huang *et al*. (2014) fit a topic model to analyst conference call transcripts and reports: to examine sell-side equity analysts' information interpretation and discovery roles. Zhao (2017) finds interesting patterns resulting from language complexity and sell-side research participation in earnings calls. Call *et al*. (2017) show that buy-side appearances on earnings conference calls are associated with subsequent decreases in sell-side coverage, lower stock returns and increases of bid-ask spreads, implied volatility and short interest.

In this chapter, we use two specific examples to show how NLP and machine learning can be used in global stock selection models. We find that NLP and machine learning-based models are uncorrelated with traditional factors, offering significant diversification benefit.

3 POS (Part-of-Speech) tagging is a powerful NLP technique that understands the meaning of a word – both its definition and its context (i.e. its relationship with adjacent and related words in a phrase) in a text.

4 PEAD is one of the most thoroughly studied market anomalies in empirical finance. However, we find that the magnitude of PEAD has declined substantially in recent years (see Wang, *et al*. 2017a). PEAD also disappears in two or three days after an earnings announcement.

In the first example, we develop a suite of analytics on management presentations. We rely on the Call Transcript Database offered by S&P Capital IQ for the underlying data. Then, we show how to apply language complexity, sentiment analysis and management personality and psychology research to conference call transcripts, to extract stock selection signals.

In the second example, we take the news sentiment data from *RavenPack*. The alpha signal from news sentiment tends to have a fairly short investment horizon (in days) and performance has decayed significantly in recent years. We demonstrate how to combine sophisticated machine learning techniques and behavioral finance with news sentiment to improve performance.

6.2. Natural language analysis of company management presentations

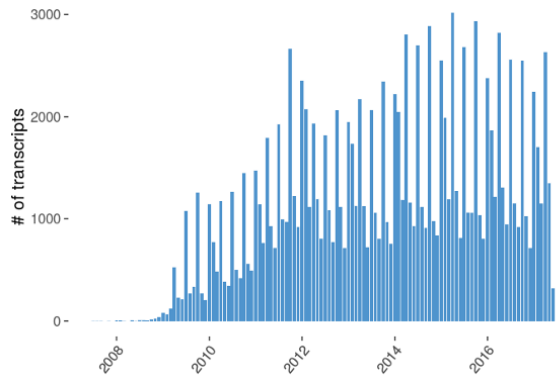
In this section, we focus on an exciting database from S&P Capital IQ – the Transcript Database, which provides current and historical call transcript data covering approximately 7,000 public companies, globally. Due to space limits, this section only provides a high-level overview. More in-depth analysis can be found in Rohal *et al*. (2018).

The Transcript Database provides textual translations of earnings calls, guidance/update, shareholder/analyst calls, analyst/investor day, M&A calls, operating result calls, fixed income calls, etc. In this chapter, we focus primarily on earnings calls as other types of calls are limited in number.

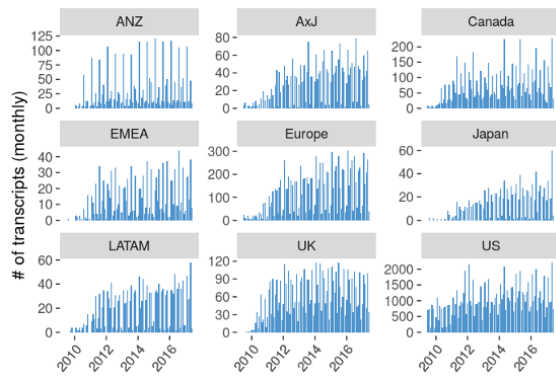
Each earnings call is typically split into two parts: "Presentation Section" and "Q&A Section". The presentation section usually includes a speech by company executives, for example CEO and CFO. The Q&A section contains conversations between company management and buy/sell-side research analysts, investors or, potentially, the media. Each sentence of the call is tagged to an executive or analyst. It also provides detailed metadata such as speaker name, speaker type and associated company for speaker. Combined with two other closely related databases, offered by S&P Capital IQ – the Professionals (background information on company executives, board members and investment professionals) and KDFE (Key Developments and Future Events) – we can generate even greater insights.

6.2.1. *Coverage*

S&P Capital IQ provides multiple copies of transcripts with differing quality at different points in time, including "proofed copy", "edited copy", "spell-checked copy" and "audited copy". We create our sample – a union of proofed and edited copies to balance timeliness and coverage. The Transcript Database begins in 2009. Currently, it covers approximately 2,500 stocks in the Russell 3000 universe (Figure 6.3a). Outside the US, coverage is somewhat limited (approximately 1,200 companies) but has improved over time (Figure 6.3b).



a) Monthly number of earnings calls

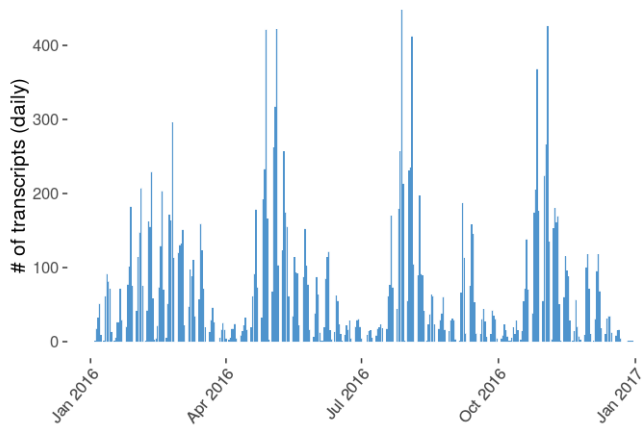


b) Number of earnings calls, by region

Figure 6.3. *Monthly coverage of call transcripts data. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

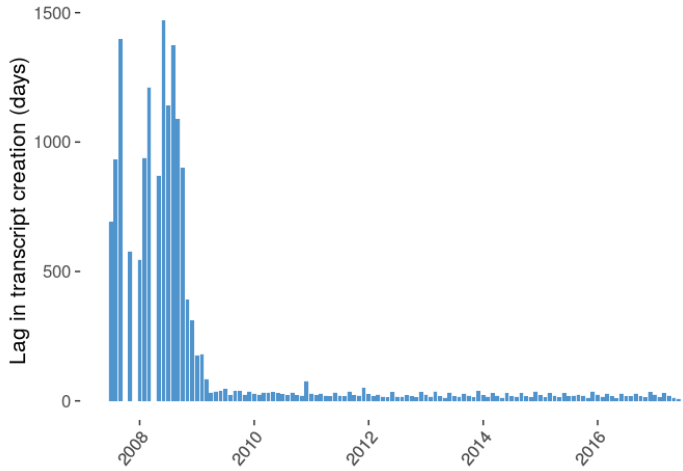
The frequency of earnings call transcripts obviously coincides with company financial reporting (quarterly, semi-annually or annually) announcements (Figure 6.4a). During the earnings season, there could be up to 400 conference calls on the same day. Attention is a limited, scarce and valuable resource. When individuals perform multiple tasks or process multiple sources of information, performance tends to suffer. Limited attention, therefore, is likely to affect how investors process information and how markets react to news (Hirshleifer *et al*. 2009). If investors' attention is indeed distracted on the earnings announcement date, the immediate market reaction is likely to be muted; therefore, the alpha contained in the transcripts is likely to last longer.

S&P Capital transforms audio/video calls into written transcripts. Part of the process can be done by computer algorithms but human intervention, such as checking, editing and verification are almost always needed. As shown in Figure 6.4b, live production of call transcripts started in mid-2009. S&P Capital IQ backfilled a few extra years of historical data. Historically, there are a few days of delay from the time when the call was conducted, to the date the transcript data became available in the database. Currently, S&P states that it aims to provide an edited copy of most transcripts within three hours after the end of a call[5].



a) Number of earnings call, 2016

5 To be conservative and avoid the potential look-ahead bias, we add a three-day lag to the call transcript data for this research.

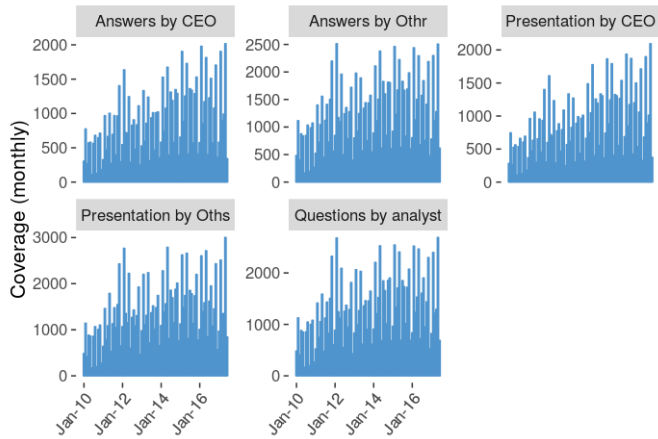


b) Delay in data processing

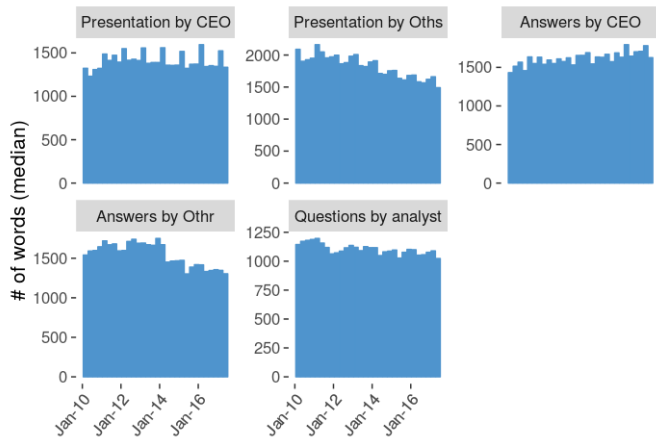
Figure 6.4. *Seasonality and processing delay in call transcript data. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Figures 6.5a and b show the average monthly number of calls and the median number of words during each call, by each section and speaker type. Obviously, CEOs participate in the vast majority of analyst calls and answer questions. Most conference calls allow analysts to ask questions and analysts do ask questions whenever possible.

During the main presentation, CEOs speak as many words as all the other executives combined. Actually, the number of words by CEOs has been steady over time, while other members of the management team have become quieter over time (Figure 6.5b). Interestingly, in the past eight years, CEOs have become more and more likely to answer questions, at the expense of other executives.



a) Number of earnings calls by participants



b) Median number of words by participants

Figure 6.5. *Coverage and number of words by section and participants. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.2.2. *Readability index and language complexity*

During the earnings season, hundreds of companies around the world can all report on the same day. Most public companies conduct analyst

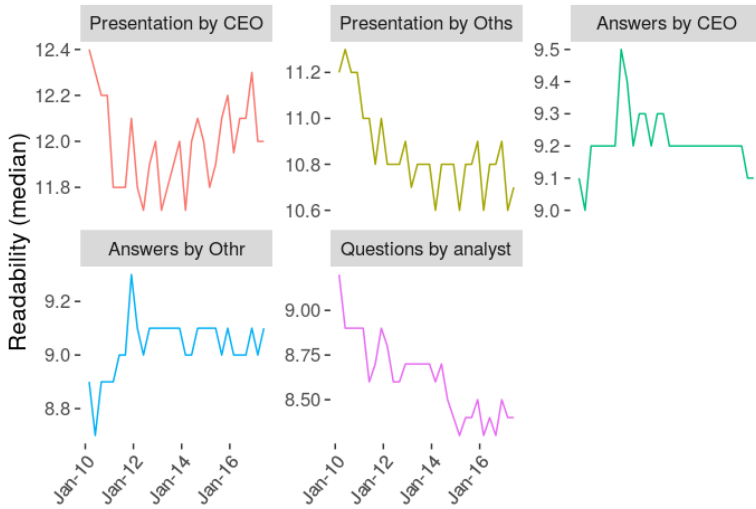
conference calls in conjunction with their press releases. Fundamental analyst read press releases, supplementary documents and the subsequent regulatory filings (with detailed financial statements, disclosures, and management discussion and analysis). Sell-side analysts, buy-side analysts/portfolio managers and the media also participate in management calls, in the attempt to gauge additional insights from the company presentation and Q&A. As shown in Hirshleifer *et al*. (2009), investors' attention is limited. The human ability to process volumes of data and information is limited. We are also often biased by our prior views of a company and overconfidence can further shadow our judgment. Therefore, investors' ability to listen to hundreds of earnings calls, to read multiple call transcripts and to derive their investment conclusions, all over a short period of time, will be highly imprecise.

One of the most basic NLP techniques is the readability test. We can gauge the language complexity using readability indices. These indices generally output a number, which approximates the grade level of education needed to comprehend the underlying text. In other words, the higher the score, the higher the complexity.

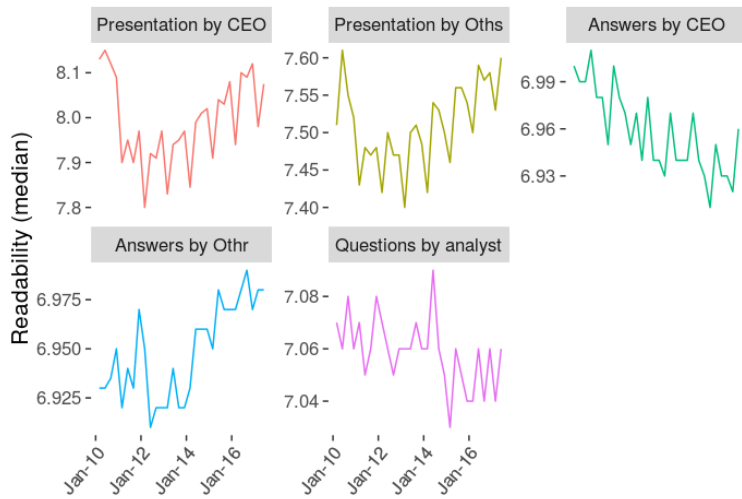
Most of these indicators are based on two factors. One factor relates to the sentence structure, for example the average number of words per sentence. The other factor relates to word structure or complexity and is usually based on either the proportion of easy words (defined by a lexicon/dictionary) or the average number of syllables per word. Some of the popular readability indices include the Automated Readability Index (Senter and Smith 1967) and the Dale–Chall Readability Score (1948, revised 1995).

6.2.2.1. *Presentations are getting fogged with complex language*

Figure 6.6 shows the median readability scores by each section/participant type. Larger scores correspond to higher education grades required to comprehend and hence, lower readability. CEO presentation has become more complex over time while the readability of analyst questions gets better in recent years. The average readability is better for the Q&A than the main presentation – spontaneously spoken language tends to be simpler than heavily scripted presentations. We also observe a strong seasonal pattern in the readability index, which coincides with the annual reporting season. Company management generally spends more time delivering their annual results than quarterly/interim updates.



a) Automated readability score



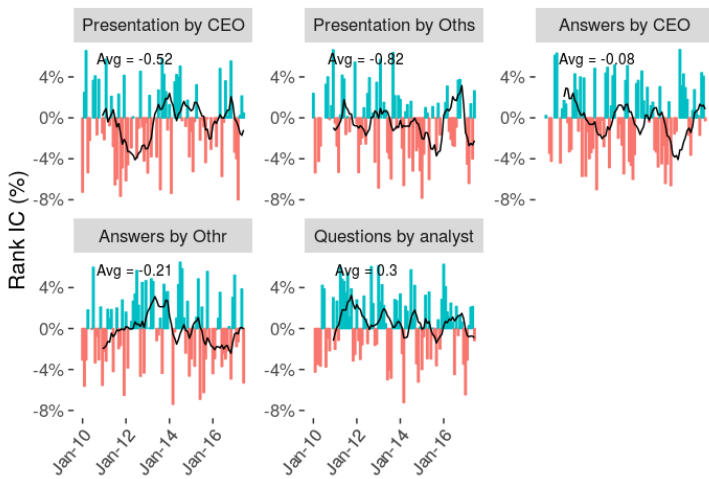
b) Dale–Chall readability score

Figure 6.6. *Median readability score, by section and speaker type. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.2.2.2. *How does the market react to complex language?*

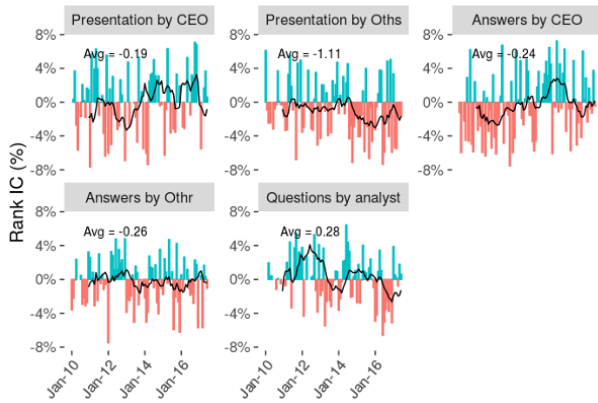
Now we turn our attention to investigating whether readability indices predict future stock returns, using the Spearman Rank IC[6]. As documented in the previous research (Rohal *et al*. 2017), most factors derived from text mining tend to be weak but persistent. The forecasting power of readability scores is mostly negative, but weak (Figure 6.7). The negative relationship between readability and future stock returns is in line with expectations. Complex language is poorly understood by investors. Furthermore, many investors associate unnecessarily complicated words with management obscurity and uncertainty.

Interestingly, more complex questions by sell-side analysts lead to slightly positive future stock returns. Elaborated questions might be indicative of a more constructive engagement between analysts and company executives.



a) Automated readability score

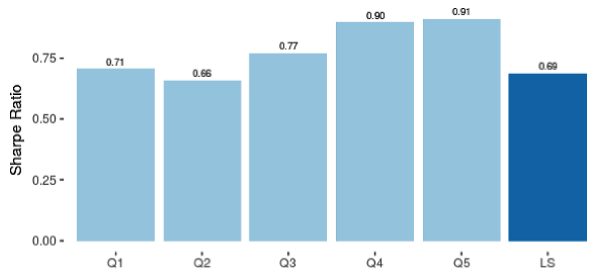
6 As a reminder, we use Rank IC to measure the predictive power of a factor in selecting/ranking stocks. It is computed as the rank correlation between the current month's signal and the following month's return, among all stocks in our investment universe.



b) Dale–Chall readability score

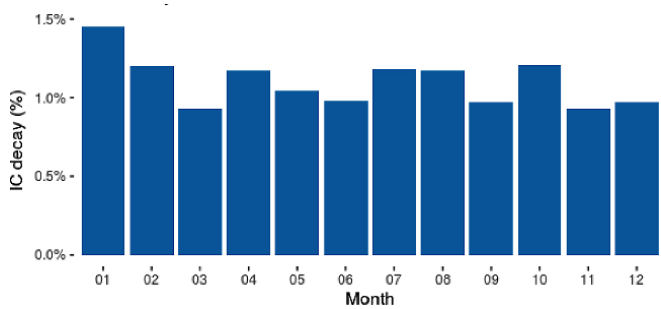
Figure 6.7. *The predictive power of readability score on future stock returns. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

We further construct a simple composite readability factor, by combining the CEO and other executive presentation sections. As shown in Figure 6.8, a simple long/short quintile portfolio[7] delivers a Sharpe ratio of 0.7x. The information decay of the signal is slow, which makes it particularly attractive for investors with long holding horizons. Textual information is sparsely distributed but signals are not crowded; therefore, returns are persistent.



a) Sharpe ratio

7 In this example, we use Russell 3000 as our investment universe. Stocks in each quintile are equally weighted. Portfolios are rebalanced monthly. Results for other countries are reported later in the chapter.



b) Rank IC decay

Figure 6.8. *Quintile portfolio performance based on executive language readability, Russell 3000 Universe. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES)*

6.2.3. *Quantifying executive personalities*

One of the cornerstones of today's fundamental research is to meet company management and attempt to get a deep understanding of how management views their own company/industry and the "styles" ("traits" or "personalities") of the executive team. Intuitively speaking, we certainly expect to find interesting relationships between executive traits and firm policies. Available academic research in this space is limited. Malmendier and Tate (2005) and Malmendier and Tate (2008) study how management overconfidence impacts a firm's investment decision and merger activity. Graham *et al*. (2013) document evidence that CEO behavior is related to measures of overconfidence, optimism and risk aversion.

There are a number of challenges to understanding how executive traits can influence firm policy, operational performance and stock prices. First, it is overly costly to engage both management and psychologists to conduct such personality tests. Senior management teams neither have the time nor interest in participating in such activities. Second, relying on human interpretation and assessment of CEO personality is neither reproducible nor objective. In a recent paper, Gow *et al*. (2015) use linguistic features extracted from conference calls and statistical learning techniques to develop a measure of CEO personality. Gow *et al*. (2015) find that management personality factors are associated with organizational strategy choices, investment and financial policies, and firm performance.

In recent years, psychology research has converged to focus on five personality factors (Goldberg 1993):

– extraversion (versus introversion): assertive, enthusiastic, proactive, persuasive, energetic;

– emotional stability (versus neuroticism): respectful, calm, resilient;

– agreeableness: forgiving, helpful, compliant;

– conscientiousness: organized, self-disciplined, efficient, persistent, hardworking;

– openness to experience: flexible, independent, intellectual, strategic, imaginative.

6.2.4. *Syntactic parser and part-of-speech (POS) tagging*

We can gauge management personality by analyzing the structure of spoken words based on formal grammar rules. In NLP terminology, this is called syntactic parsing. A syntactic parser reads an input sentence and describes its grammatical structure. The parser usually returns a graph of word–word relationships, aiming to extract the reasoning from the underlying text. These parsed trees are useful for grammar checking or extracting meaning from news articles. More importantly, parsed trees are used for semantic analysis and hence, play an important role in chat bots.

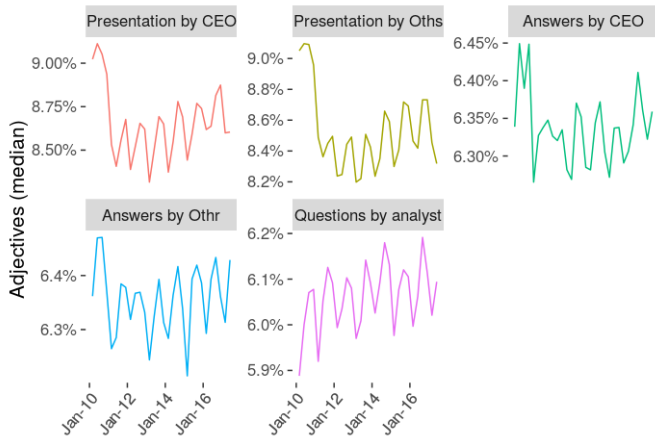
One aspect of semantic analysis is part-of-speech (POS) tagging. It is the process of tagging words or tokens to their respective part of speech classes, such as nouns, verbs and adjectives. These classes are known as lexical categories or parts of speech.

6.2.4.1. *Use of adjectives/adverbs*

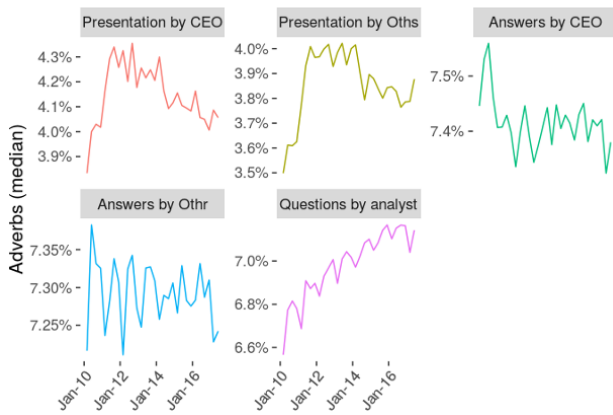
In language, adverbs modify verbs, while adjectives modify other nouns and pronouns. In presuppositions, even if you disagree with the adjective or adverb in the sentence (e.g. “This quarter’s earnings were extremely strong.”), you still accept what is modified (e.g. “This quarter’s earnings were strong”). Because adjectives and adverbs are less objective, the ratio of these words in sentences is a crude way of measuring personality.

We can simply count the occurrence of adjectives or adverbs in a call transcript as a crude way to assess the subjectivity of the language spoken by

the participants. Figure 6.9 shows the percentage of adjectives and adverbs in the call transcript database. There is no clear trend from the presentation or the answers from company executives. We see that research analysts are moving away from simple to-the-point questions to more subjective language with a higher percentage of adjectives and adverbs.



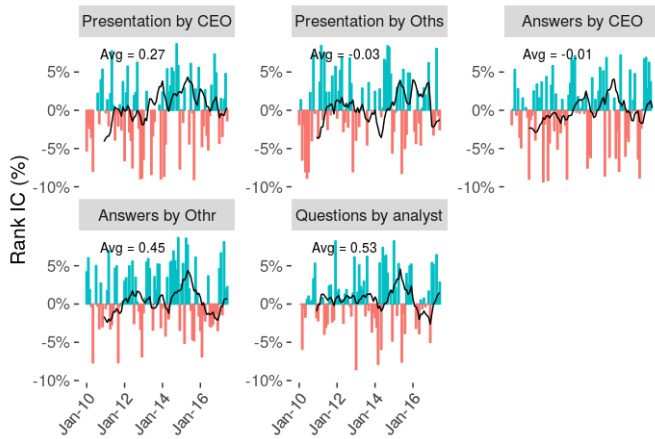
a) Percentage of adjectives



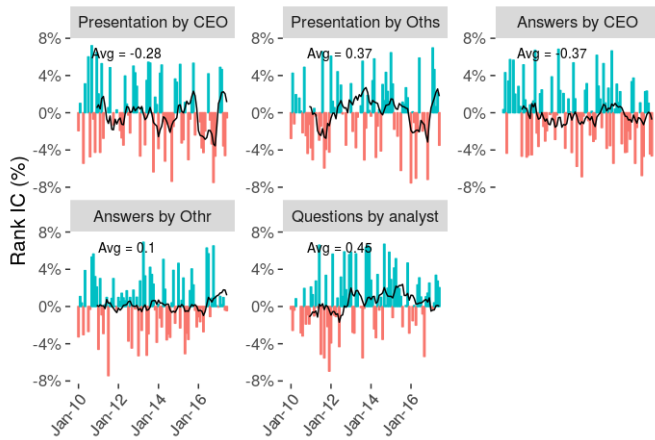
b) Percentage of adverbs

Figure 6.9. *Part-of-speech distribution. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

These crude measures of personality features are not particularly useful in predicting future stock returns (Figure 6.10a and b). However, we do note that more descriptive/subjective answers by CEOs are negatively correlated with subsequent stock performance (Figure 6.10b).



a) Rank IC of adjectives

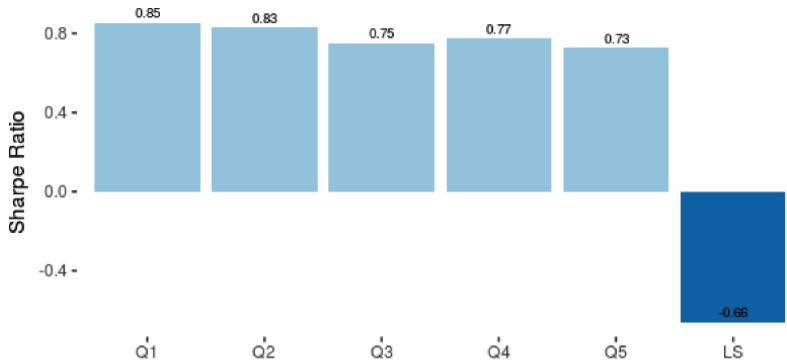


b) Rank IC of adverbs

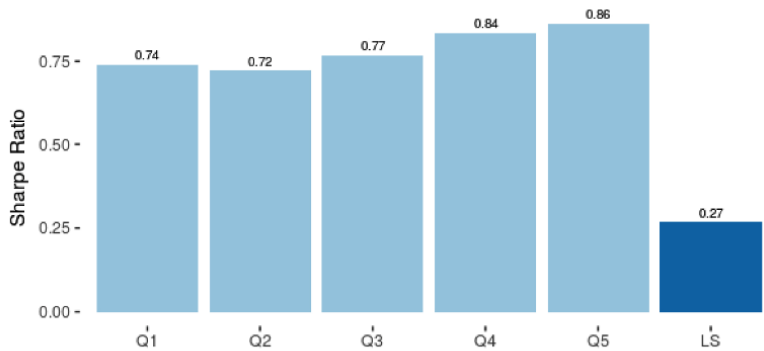
Figure 6.10. *Percentage of adjectives/adverbs, Rank IC. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.2.4.2. *Use of pronouns*

Another basic feature of personalities is the use of first person singular (I, me, my, mine) or second person plural (we, us, our, ours) pronouns in spoken language. This is a subtle hue on the character of the executives as team player. The performance of the signal is again modest but the returns are in the intuitive direction. Higher reference to self/first person is negative, while more use of team/second person pronouns is positively associated to future company performance (Figure 6.11).



a) Singular first person pronouns



b) Plural first person pronouns

Figure 6.11. *Use of pronouns, quintile portfolio, Sharpe ratio. (Sources: Bloomberg Finance LLP, FTSE Russell, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES)*

6.3. Extracting long-term signal from news sentiment data[8]

News and sentiment-related data has generated considerable attention from academia. In one of the seminal papers, Tetlock *et al*. (2008) study daily data from the Wall Street Journal and find that high media pessimism predicts downward pressure on stock price the following day. Sprenger and Welpe (2010) extract news data from an online stock forum and study its impact on asset pricing. The authors classify events in a few broad categories and then apply sentiment analysis. They find that aggregate news events by topic contain little information about future asset returns. Rather, news sentiment does have the predictive power of future stock returns.

Boudoukh *et al*. (2014) find that stock price reversals occur on "no news" days, while days identified as "news days" show the opposite effect – namely a strong degree of continuation. More recently, Huynh *et al*. (2015) suggest that the performance of the standard price momentum factor can be enhanced by adding news signals derived from Thomson Reuters News Analytics.

Lastly, Kearney *et al*. (2013) provide a comprehensive literature review of news and sentiment-related research. In summary, most of these academic studies suffer from limited data coverage and short data history. Furthermore, most existing research seems to suggest that the predictive power of textual data is only relevant for a short horizon, usually consisting of a few days.

6.3.1. *Introducing RavenPack data*

RavenPack Analytics transforms unstructured news and social media data into systematic indicators. It can be used as a source for alternative alpha, as well as to manage event risk. RavenPack Analytics allows investors to incorporate news, social media and sentiment into their investment process in real time. In addition, news events on natural disasters, political upheaval,

8 Due to space limit, this section only provides an overview of the topic. More detailed analysis can be found in Rohal, *et al*. (2019).

regulatory uncertainties, and many others, can be precursors to volatility spikes at both individual company and aggregate market levels.

Figure 6.12 shows a snapshot of the RavenPack database, with a few selected data items:

– *time stamp*: every news item is properly time stamped in milliseconds[9];

– *entity name*: each news item is tagged to potentially many entities. An entity can be a company, a place, an individual, a country, a currency, etc. Entity recognition is one of RavenPack's key strengths. Each entity has its own unique identifier. For the purpose of this research, we mostly examine publicly traded companies, which are properly mapped into our global equity database;

– *relevance*: RavenPack also provides a relevance score for each of the related entities. The relevance score is always between 0 (completely unrelated) to 100 (very relevant to the entity). In our experience, it is critical to build our signals only on these highly relevant news articles[10];

– *Novelty* (Similarity Days): RavenPack provides a granular number which indicates the number of days since a similar event was detected. Values range between 0 and 365. A value of 365 means that the most recent similar story may have occurred 365 or more days previously. For many news stories, there are typically multiple updates, even from the same news provider (e.g. Dow Jones Newswires). The market normally only reacts to the first (few) releases of the news. The market impact on the subsequent updates tends to be far more modest;

– *Sentiment*: RavenPack provides a suite of pre-computed sentiment measures based on its proprietary NLP algorithms:

- the headline RavenPack sentiment signal is called ESS (Event Sentiment Score), a score between 0 and 100[11]. It is based on RavenPack's NLP algorithms and-extensive database of time an sensitive information about each entity,

9 We only show date in this example. In this research, we focus on the long-term implications of news stories.
10 A score of 75 and above is normally considered as significant.
11 Zero represents the most negative sentiment, while 100 means the most bullish tone.

- NIP (News Impact Projections) measures the impact of a particular news item on stock volatility in the following two-hour window,

- MCQ (Multi Classifier for Equities) is only applicable towards the most relevant companies mentioned in a story. MCQ is constructed similarly to the ESS, with some noticeable differences,

- CSS (Composite Sentiment Score) sentiment is trained on how the market reacted to similar words/phrases in the past, using intraday tick data,

– *event category*: RavenPack automatically detects key news events and identifies the role played by the entity. There are four layers of event category:

- *topic* is the highest level of event taxonomy;

- there are multiple *groups* within a topic category;

- the third event level is *type*;

- the most granular level is the *sub-type*.

We further extend these signals by computing aggregate sentiment, abnormal volume and market reaction, using a variety of rolling windows. We measure abnormal volume as the deviation of news volume from its long-term trend and market reaction to news announcements, as the excess returns during the event week.

In RavenPack Analytics, news is received and processed through an automated real-time data stream made up of several parallel software components. Events in RavenPack's taxonomy are defined using thousands of proprietary template programs and POS tagging. RavenPack systematically categorizes stories into a simple set of themes (i.e. Topics, Groups, Types and Sub-Types) which is essential for investment research. Next, the context in which the entities are mentioned is analyzed to determine what role they played in the story. Events can have more than one participant, with only one being the principal.

timestamp_utc	entity_name	relevance	ess	similarity_days	topic	grp	type	sub_type	headline	css	nip	mcq
05/01/2018 15:44:09	Aetna Inc.	100	-0.24	365	business	revenues	revenue	down	Aetna Inc. Q1 Total Revenues USD 15.34B Vs USD 15.49B	0	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.44	365	business	earnings	pretax-earnings-expectations	above-expectations	Aetna Inc. Reports Q1 Pre-Tax Income Adjusted USD 1.34B Vs Consensus USD	0	0	0
05/01/2018 15:44:09	Aetna Inc.	100	-0.27	365	business	revenues	revenue	below-expectations	Aetna Inc. [illegible] 34B Vs Consensus USD 15.35B	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	-0.21	365	business	earnings	pretax-earnings	down	Aetna Inc. Q1 [illegible] 34B Vs USD 1.45B	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.86	365	business	earnings	pretax-earnings	up	Aetna Inc. Q1 [illegible] 3B Vs USD -628.00M	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.46	365	business	earnings	earnings-expectations	above-expectations	Aetna Inc. Reports Q1 Net Profit Adjusted USD 973.00M Vs Co	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.4	365	business	earnings	earnings	up	Aetna Inc. Q1 Net Profit Adjusted USD 973.00M Vs USD 939.00	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	-0.26	365	business	earnings	earnings-per-share-expectations	below-expectations	Aetna Inc. Reports Q1 EPS-Non-GAAP USD 2.82 Vs Consensus	[illegible]	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.4	365	business	earnings	earnings-per-share	up	Aetna Inc. Q1 EPS-Non-GAAP USD 2.82 Vs USD 2.54	[illegible]	[illegible]	[illegible]
05/01/2018 15:44:09	Aetna Inc.	100	-0.23	365	business	earnings	ebit-expectations	below-expectations	Aetna Inc. Reports Q1 EBIT Adjusted USD 1.30B Vs Consensus USD 1.33B	0	0	0
05/01/2018 15:44:09	Aetna Inc.	100	0.37	365	business	earnings	ebit	up	Aetna Inc. Q1 EBIT Adjusted USD 1.30B Vs USD 1.26B	0	0	0
05/01/2018 15:44:09	Aetna Inc.	100	-0.2	365	business	earnings	ebitda	down	Aetna Inc. Q1 EBITDA Adjusted USD 1.50B Vs USD 1.63B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	0.27	365	business	earnings	ebitda-expectations	meet-expectations	Aetna Inc. Reports Q1 EBITDA USD 1.52B Vs Consensus USD 1.52B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	-0.22	365	business	earnings	ebitda	down	Aetna Inc. Q1 EBITDA USD 1.52B Vs USD 1.70B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	-0.2	365	business	earnings	ebita	down	Aetna Inc. Q1 EBITA USD 1.42B Vs USD 1.54B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	0.43	365	business	earnings	ebit-expectations	above-expectations	Aetna Inc. Reports Q1 EBIT USD 1.38B Vs Consensus USD 1.37B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	-0.2	365	business	earnings	ebit	down	Aetna Inc. Q1 EBIT USD 1.38B Vs USD 1.45B	0	0	0
05/01/2018 15:44:10	Aetna Inc.	100	0.38	365	business	dividends	dividend	up	Aetna Inc. Q1 Dividend Per Share USD 0.51 Vs USD 0.50	0	0	0
05/01/2018 15:44:10	Aetna Inc.	[illegible]	[illegible]	365	business	earnings	earnings-expectations	above-expectations	Aetna Inc. Reports Q1 Net Profit Reported USD 930.75M Vs Consensus USD 92	0	0	0
05/01/2018 15:44:10	Aetna Inc.	[illegible]	[illegible]	365	business	earnings	earnings	up	Aetna Inc. Q1 Net Profit Reported USD 930.75M Vs USD -381.00M	0	0	0
05/01/2018 20:05:43	Deutsche Bank AG	[illegible]	[illegible]	365	business	stock-prices	stock-price	loss	10:33940266816	NA	0	0
05/01/2018 20:05:43	Deutsche Bank AG	99	-0.58	147	business	analyst-ratings	analyst-ratings-change	negative	Deutsche Bank Aktiengesellschaft (DB) Hit 52-Week Low	0	-0.18	0
05/01/2018 20:07:04	Virgin Australia Holdings Ltd.	100	0.49	365	business	partnerships	partnership		Virgin Atlantic and Virgin Australia Expand Codeshare Relationship	0.04	-0.22	1
05/01/2018 20:07:04	Virgin Atlantic Airways Ltd.	100	0.49	365	business	partnerships	partnership		Virgin Atlantic and Virgin Australia Expand Codeshare Relationship	0.04	-0.22	1
05/01/2018 20:09:34	Alliance Bernstein Holding L.P.	100	0	365	business	assets	facility	relocation	AllianceBernstein Move to Nashville to Begin Later This Year -- Sources	0	-0.22	0
05/01/2018 20:09:34	Alliance Bernstein Holding L.P.	100	0.06	365	business	assets	headquarters-change		[illegible]lianceBernstein Holding LP to Move Headquarters and Most Staff to Nashvil	0	-0.34	0
05/01/2018 20:07:12	Western Union Co.	100	0.59	365	business	equity-actions	buybacks		[illegible]estern Union Reports First Quarter Results	0	0.12	0
05/01/2018 20:09:45	Sunesis Pharmaceuticals Inc.	100	0	365	business	investor-relations	conference-call		[illegible]:33940293349	NA	0	0
05/01/2018 20:10:00	Five9 Inc.	100	0.53	365	business	labor-issues	executive-appointment		Five9 Appoints Industry Veteran Rowan Trollope As New CEO >FIVN	0.1	0.04	1
05/01/2018 12:22:40	Lightbridge Corp.	100	0	153	business	marketing	conference		Lightbridge Corporation to Present at the 3rd Annual Disruptive Growth and H	0.02	0.18	1
05/01/2018 12:26:16	FedEx Corp.	100	0.39	365	business	products-services	business-contract		BRIEF-Plug Power and Workhorse Provide FedEx With Fuel Cell-Powered Elec	0.02	-0.48	0

Multiple updates of the same earnings release

Multiple sentiment measures

Including non-US

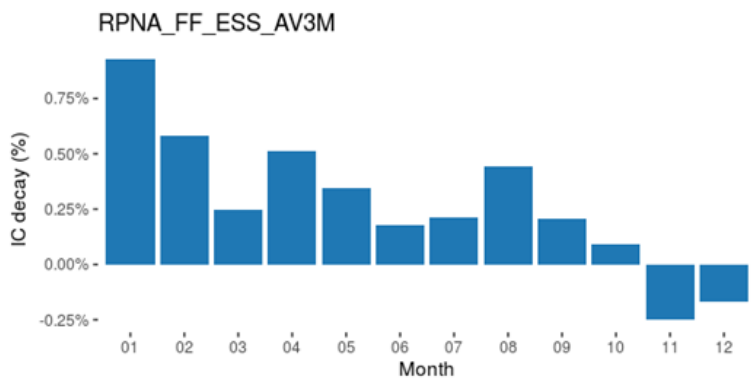
Including non-earnings/revenue news

Figure 6.12. *RavenPack data sample snapshot. (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES)*

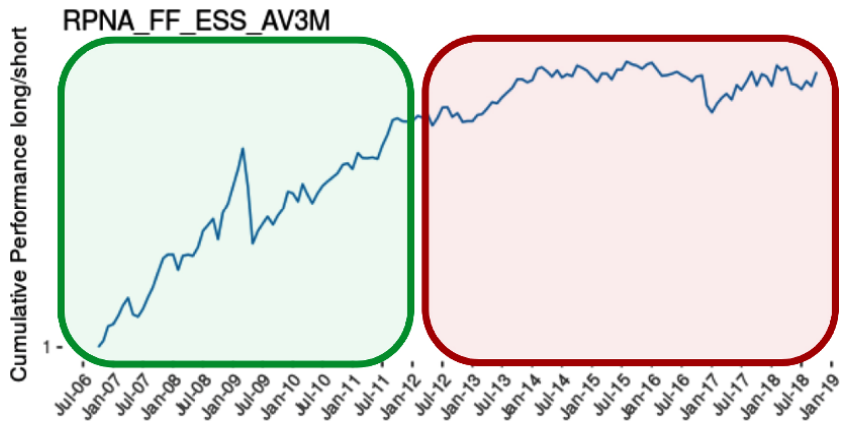
The average overall latency of a processed news story at any time is at 300 milliseconds. The latency is defined as the average time it takes a story to get through the system from the time RavenPack receives it, to the time at which it leaves the Event Distribution Server at the RavenPack Data Center. RavenPack Analytics delivers sentiment analysis on more than 192,000 entities in over 130 countries and covers over 98% of the investable global market. It provides exhaustive coverage by entity type with more than 100,000 places, 43,000 companies, 1,000 products and more than 100 nationalities, currencies and commodities.

6.3.2. *The challenges of using news sentiment signals in stock selection*

Figure 6.13 shows the historical performance of the headline RavenPack sentiment signal (ESS) in the US (i.e. the Russell 3000 universe). In this case, the sentiment signal is computed using a three-month moving average. Similar to our previous experience, the performance of the naive sentiment factor is not particularly strong. At the monthly frequency, the sentiment factor loses significant predictive power, especially since 2012.



a) IC decay



b) Cumulative performance, L/S quintile portfolio

Figure 6.13. *Event sentiment score (ESS), three-month aggregate factor performance (US). (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.3.3. *How do investors react to news?*

The traditional way of using news sentiment in investing mostly relies on the assumption that investors under-react to news. When there is positive (negative) news on a company, most investors are unlikely to immediately act on the news and buy (sell) the stock. Rather, investors want further confirmation with more developments. Since the information diffusion process takes time, it forms a price continuation process. As a result, slower moving investors (i.e. non-high frequency traders) can still watch news and benefit from news analytics.

With the rapid development of social media, mainstream news outlets have also been forced to make information readily available at low or no cost. News analytics and sentiment analysis have also been widely developed and are now available from multiple data vendors. As news sentiment data is increasingly accessible to investors, the under-reaction to news assumption is less likely to hold. Indeed, as we will show in this section, on many occasions, investors actually tend to overreact to news.

We would like to hypothesize that how investors react to news heavily depends on the nature of the news. The vast majority of news is associated with specific events. As discussed in the previous section, RavenPack tags each news story to an entity and event category. There are four levels of events – Topic, Group, Type and Sub-Type. As shown in Figure 6.14, at the Group level, news stories are overwhelmingly dominated by earnings and revenue-related events. In fact, earnings, revenue and stock-price related stories account for almost 90% of events. As a result, our previously discussed sentiment factor is highly skewed towards these few categories. Most of such news stories are already captured by various traditional analyst estimate and price momentum/reversal factors and therefore may not provide much additional information. The underrepresented, low frequency events, such as regulatory, legal and labor-issues can be quite relevant to asset returns and are generally not well captured by traditional stock selection factors.

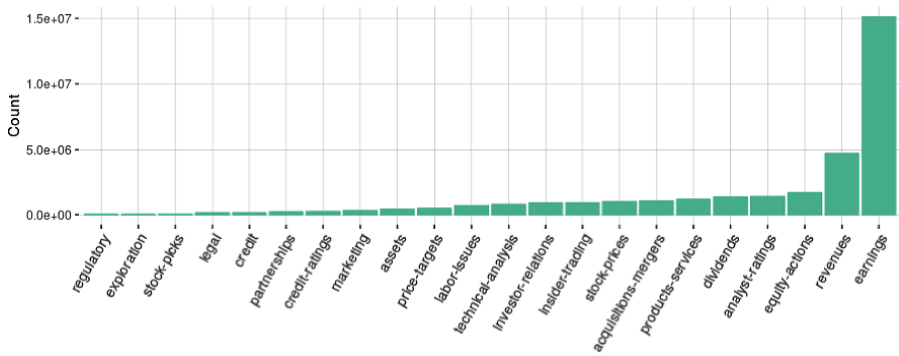


Figure 6.14. *Frequency count by RavenPack event groups (level II). (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES)*

6.3.4. *The interaction of news, corporate events and investor behavior*

To assess the way that investors react to news, in this section we conduct a series of event studies. We argue that there are two extreme categories of corporate events. The first category includes the slow diffusion non-headline news that investors read, but are unlikely to take immediate action. At the

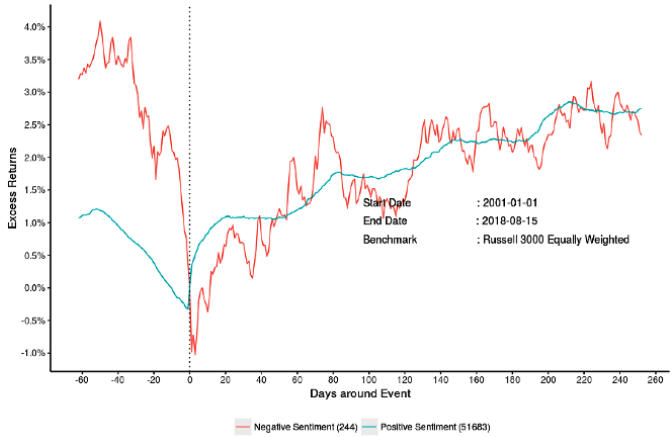
other extreme, there are major market-moving headline news items such as FDA drug-related news, unexpected regulatory enquiries and litigations, etc. Investors typically overreact to such news.

The difficulty of understanding market reactions to a given news story is that, around the news announcement date, there are often other company-specific factors impacting a stock's return. By combining many companies that underwent a similar type of events at different points in time, these company-specific idiosyncrasies are largely eliminated. Therefore, we can better extract the common patterns around event date. Event studies are helpful to understand investor behavior and therefore design more effective investment strategies. For example, if we see a significant and persistent negative (positive) price trend leading to negative (positive) news, it may indicate information leakage prior to public release of news. A sustained price drift in the same direction, post the event day, can be attributed to market under-reaction and slow information diffusion, which presents profitable investment opportunities. Similarly, the market may overreact to news which leads to price reversal, post the event date.

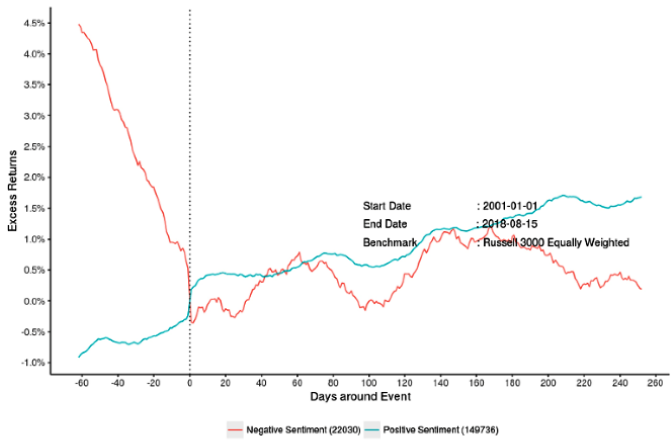
6.3.4.1. *Slow diffusion and under-reaction to news*

The first category of slow diffusion and investor under-reaction mostly relates to small positive news that investors view favorably, but are unlikely to buy the stock immediately. We use two specific examples – share buybacks and dividend announcements – to demonstrate.

Companies tend to announce share buybacks when the management believes their own stocks to be undervalued. Most buyback news is updates on the status of stock repurchase programs and therefore has a positive sentiment. Buybacks are typically initiated on the back of some short-term underperformance. The pre-announcement period return is mostly negative, regardless of whether the news is positive or negative. Similarly, the announcement-day return and post-announcement drift are positive, on average – for both positive and negative news stories (Figure 6.15a). Investors typically welcome positive buyback news but tend to under-react to such announcements. As a result, the post-announcement drift for such news is consistently positive, lasting for about a year, post events. The post-announcement drift for positive development on dividend announcements resembles buybacks (Figure 6.15b).



a) Buyback



b) Dividend

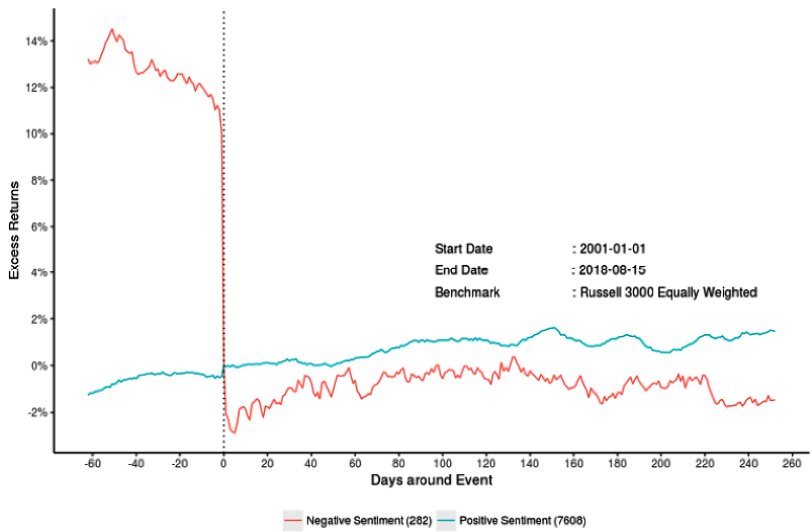
Figure 6.15. *Event study for buyback and ownership (US). (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.3.4.2. *Major headline and market overreaction*

The second category of news mostly concerns major market-moving news – typically negative announcements. In such cases, investors tend to overreact on the event day, which leads to some mean-reversal pattern after the event.

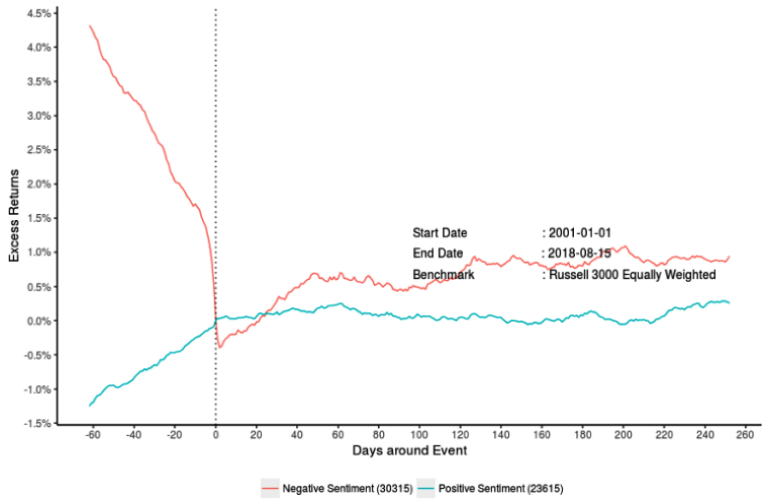
In the first example, we use clinical trials (one of the pre-defined events by RavenPack) for BioPharma companies[12]. Clinical trials for pharmaceutical companies involve large capital investment and long multi-year development cycles; therefore, they are typically market-moving events. A negative outcome of a clinical trial on average leads to a large correction of –10% on the announcement day (Figure 6.16a). On average, stock prices recover some of the initial losses in the two months post event. On the other hand, positive clinical trial outcomes are much less headline-grabbing, with small announcement day return, but lead to more persistent post-announcement drift.

Credit analysts, due to their conservative nature, tend to be more reactive than proactive. They typically downgrade (upgrade) after a period of share price underperformance (outperformance). The market reacts to negative news and downgrades far more severely than upgrades. In the end, a going concern credit rating causes more damage to shareholders than a credit upgrade. Equity investors exhibit a classic "sell first and think about it later" behavioral pattern, in that the market overreacts to negative news, which leads to a reversal pattern post negative credit events (Figure 6.16b).



a) Clinical trial

12 For investors, BioPharma stocks are typically liquid with wild swings, offering tremendous alpha opportunities.



b) Credit rating change

Figure 6.16. *Event study for sector specific event (US). (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.3.5. *A machine learning approach to extract event-based sentiment*

The traditional way of using news sentiment signals ignores the fact that not all news stories are the same – different types of events may have very different implications for future asset returns. Similarly, conventional event studies may not necessarily consider the positive/negative nature of each event. Furthermore, most investors do not trade on corporate events on their own. In this section, we attempt to combine news, sentiment, event, news volume and market behavioral biases together – all in a systematic way, using cutting edge machine learning techniques.

As discussed in the previous sections, there are complex interactions among news event categories, sentiment, volume, volatility and market impact. In this section, we use two advanced machine learning techniques to

untangle the complex relationship among the following variables, and build a predictive model of future stock returns:

– *news event*: we rely on RavenPack's news event classification algorithm, by focusing on Group (Level II) and Type (Level III);

– *news sentiment*: we use the three pre-defined sentiment measures – ESS, MCQ and CSS – all computed over different rolling windows;

– *news volume*: we compute several news volume factors, i.e. frequency of news stories over various look-back windows. New volume factors are all adjusted for size;

– *volatility impact*: RavenPack offers a pre-defined NIP factor, i.e. predicted impact on stock volatility;

– *market behavioral bias*: to measure how the market reacts to each event, we compute a series of event-day return (and abnormal trading volume) factors. Event-day returns (and abnormal trading volumes) are typically calculated on a five-day window, i.e. two days before and two days after each event. In addition, since many news events are sporadic, arriving at low frequency, we further aggregate our signals over a few rolling windows (one, three and six months).

6.3.5.1. *Investment universe*

We divide the global equity market into three universes:

– US (Russell 3000);

– developed ex-US: Canada, Europe, UK, Developed Asia (HK and Singapore), ANZ (Australia and New Zealand);

– emerging markets.

RavenPack only covers English language news, therefore the coverage is not particularly strong in certain countries. The three-region grouping above gives us a large enough sample to train our machine learning models.

6.3.5.2. *A machine learning approach*

As a demonstration, we train two machine learning models for each Event Group (and Type), for each of the above three investment universes. In both models, the target variable is stock returns in the following quarter,

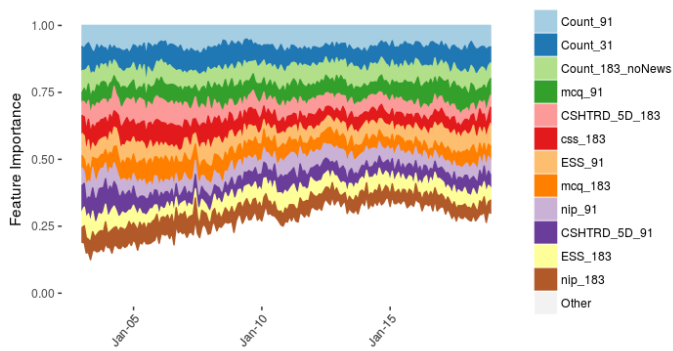
while the factors/features used in the model are news sentiment and event counts:

– *a linear LASSO (Least Absolute Shrinkage and Selection Operator) model*: unlike OLS (ordinary least squares) regression, LASSO is essentially a shrinkage-based method. It improves forecasting accuracy by shrinking large regression coefficients to reduce overfitting;

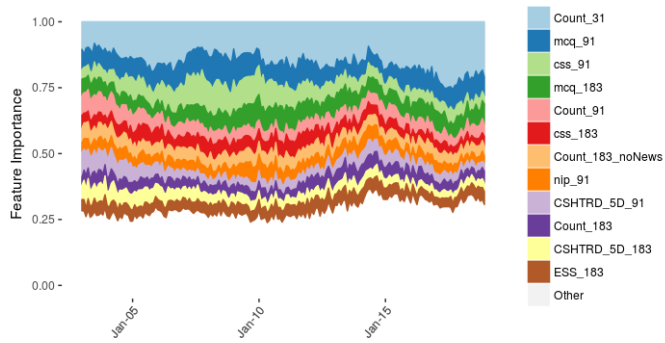
– *a nonlinear XGBoost model*: XGBoost is an efficient implementation of the GBM (Gradient Boosting Machine) model. Gradient boosting is a machine learning technique for regression and classification problems, using an ensemble (e.g. averaging) of many underlying models – typically decision trees. Binary decision trees can effectively capture nonlinear relationships in data, while taking an average (i.e. ensemble) of multiple decision trees generally improves prediction accuracy considerably.

An equally weighted LASSO and XGBoost model are further created for each event type, i.e. event-based sentiment factors.

As an example, Figure 6.17 shows the chosen factors by the XGBoost algorithm (i.e. feature importance) for legal and revenue events, respectively. The relative importance of factors varies greatly by event types. However, we do observe a robust mix of event count, sentiment, volatility and market impact factors.



a) Legal



b) Revenue

Figure 6.17. *Feature importance for legal and revenue event types (US). (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.3.5.3. *Model performance*

Figure 6.18 shows the performance (as measured by rank IC) and coverage of our event-based sentiment factor for each event type in the US. Both coverage and performance vary greatly across event types. For example, layoffs and same-store sales have low coverage but strong performance. On the other hand, insider-trades and ownership have broader coverage but poor cross-sectional predictive power. Earnings- and revenue-related events have both strong coverage and performance.

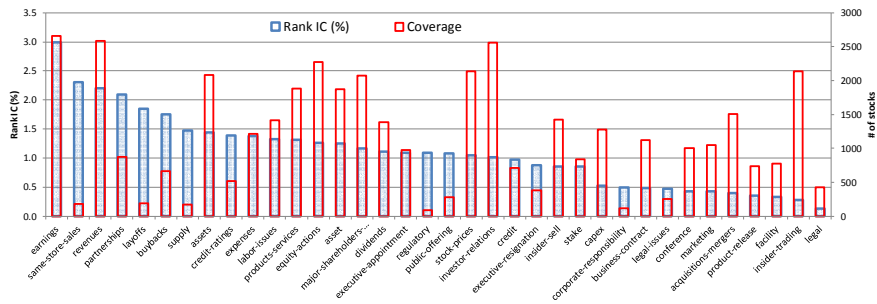


Figure 6.18. *Performance of event-based sentiment factors in the US. (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Furthermore, as shown in Figure 6.19, our event-based sentiment factors outperform the off-the-shelf sentiment signals (e.g. ESS, MCQ, CSS) considerably.

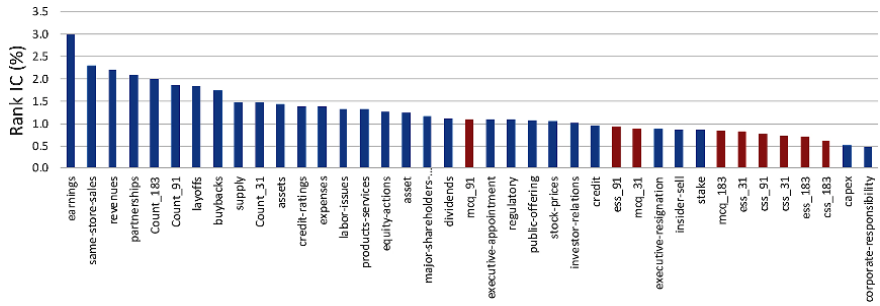


Figure 6.19. *Rank IC for each event groups/types composite and basic aggregate sentiment factors. (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.3.6. *Welcome to NICE (News with Insightful Categorical Events)*

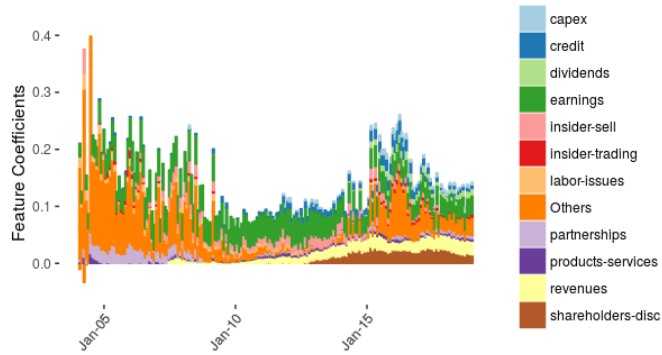
Finally, we use the elastic net machine learning algorithm to combine our suite of event-based sentiment factors together. The final model is called the NICE[13] (News with Insightful Categorical Events) model.

Elastic net is essentially a regularization technique combining the Ridge (using sum of squared coefficients penalty) and the LASSO (using sum of absolute coefficients penalty) algorithms. The LASSO regularization shrinks many regression coefficients to zero, leaving us with just a couple of events linked to earnings and revenue. Although these event types have strong performance and great coverage, they are likely to be highly correlated with conventional accounting-based factors. On the other hand, the Ridge regression shrinks the regression coefficients towards zero, but not necessarily to zero.

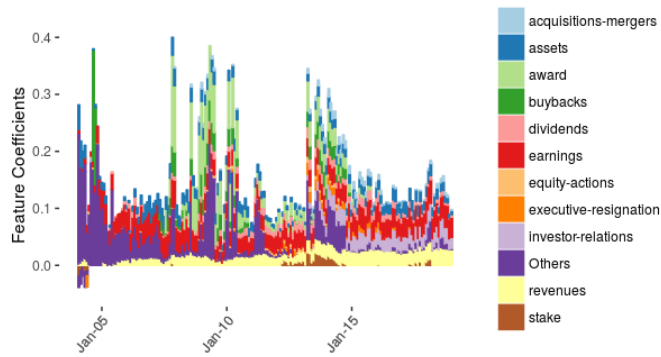
As shown in Figure 6.20, by blending the Ridge regression into the LASSO, the elastic net algorithm maintains a more balanced allocation among different event categories. Revenue- and earning-related event types still have the largest weights. Coefficients for other event types differ

13 We coin the model the French city Nice (pronounced as /ni:s/ in IPA), rather than the English word nice (pronounced as /naɪs/ in IPA).

significantly from one region to another. Shareholder disclosure, partnership, products-services and labor-issues are prominent in the US (Figure 6.20a). On the other hand, for developed markets ex-US, investor-relations, assets, award and buyback events contribute, to a meaningful extent, to the NICE model (Figure 6.20b).



a) US

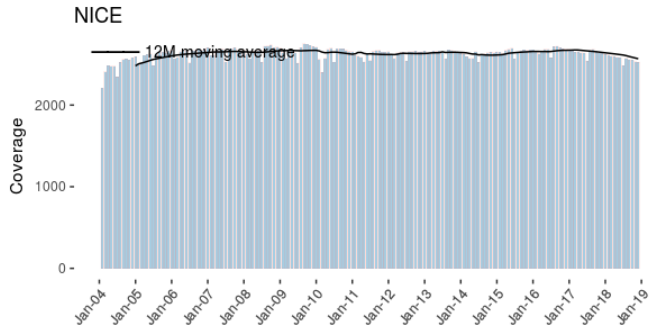


b) Developed markets ex-US

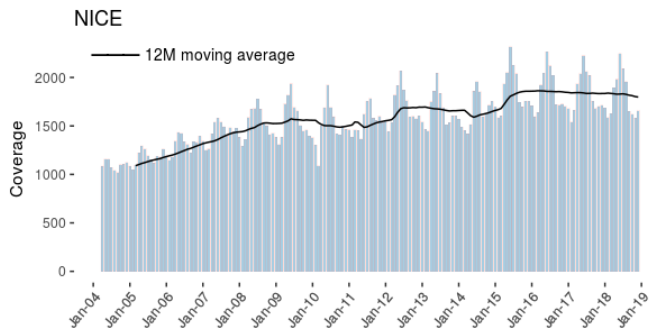
Figure 6.20. *Feature importance for NICE composite. (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

As shown in Figure 6.21a, the NICE model has nearly complete coverage of the Russell 3000 universe in the US, and covers 1,500 stocks in both developed markets ex-US (Figure 6.21b) and emerging markets

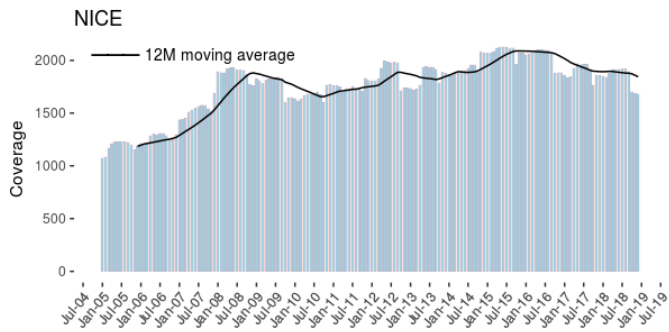
(Figure 6.21c). The NICE model is particularly effective in predicting future stock returns in Asia ex-Japan, Europe, US and emerging EMEA, while it struggles in Canada and Japan (Figure 6.21d).



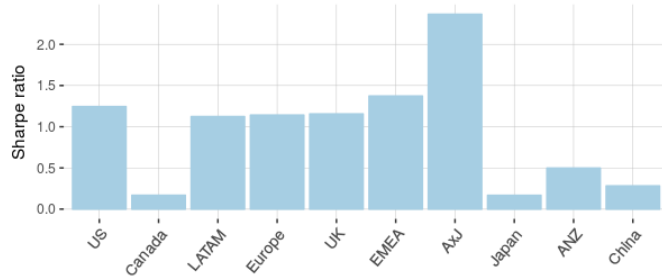
a) Coverage, US



b) Coverage, developed markets ex-US



c) Coverage, EM



d) Sharpe ratio (L/S quintile portfolio)

Figure 6.21. *NICE model performance. (Sources: Bloomberg Finance LLP, FTSE Russell, Google, IDC, RavenPack, S&P Capital IQ, Thomson Reuters, SEC, Wolfe Research Luo's QES). For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

6.4. References

Ahmad N. and Zinzalian A. (2010). Predicting stock volatility from quarterly earnings calls and transcript summaries using text regression. Report, Stanford NLP, California.

Antweiler W. and Frank M. (2002). Is all that talk just noise? The information content of internet stock message boards. *Journal of Finance*, 59(3), 1259–1294.

Boudoukh J., Feldman R., Kogan S. and Richardson M.P. (2013). Which news moves stock prices? A textual analysis. SSRN Working Paper, NBEK, Massachusetts.

Call A., Sharp N. and Shohfi T. (2017). Implications of buy-side analysts' participation in public earnings conference calls. SSRN Working Paper.

Chall J.S. and Dale E. (1995). *Readability Revisited*. Brookline Books, Boston.

Dale E. and Chall J. (1948). A formula for predicting readability. *Educational Research Bulletin*, 27, 11–20.

Das S. and Chen M. (2001). Yahoo! For Amazon: Sentiment parsing from small talk on the web. SSRN Working Paper, SSRN.

Goldberg L.R. (1993). The structure of phenotypic personality traits. *American Psychologist*, 48, 26–34.

Gow I.D., Kaplan S.N., Larcker D.F. and Zakolyukina A.A. (2015). CEO personality and firm policies. Working Paper, NBER, Massachusetts.

Graham J.R., Campbell R.H. and Puri M. (2013). Managerial attitudes and corporate actions. *Journal of Financial Economics*, 109, 103–121.

Hirshleifer D., Lim S.S. and Teoh S.H. (2009). Driven to distraction: Extraneous events and underreaction to earnings news. *Journal of Finance*, 60, 2289–2325.

Huang A., Lehavy R., Zang A.Y. and Zheng R. (2014). A thematic analysis of analyst information discovery and information interpretation roles. SSRN Working Paper.

Huynh T.D. and Smith D.R. (2015). Stock price reaction to news: The joint effect of tone and attention on momentum. SSRN Working Paper.

Kearney C. and Liu S. (2013). Textual sentiment in finance: A survey of methods and models. SSRN Working Paper.

Li F. (2006). Do stock market investors understand the risk sentiment of corporate annual reports? SSRN Working Paper.

Loughran T. and McDonald B. (2011). When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. SSRN Working Paper.

Luo Y., Zhong J. and Alvarez M. (2019). The future of active management. *Wolfe Research Luo's QES*.

Malmendier U. and Tate G. (2005). CEO overconfidence and corporate investment. *Journal of Finance*, 60, 2661–2700.

Malmendier U. and Tate G. (2008). Who makes acquisitions? CEO overconfidence and the market's reaction. *Journal of Financial Economics*, 89, 20–42.

Price S.M., Doran J., Peterson D. and Bliss B. (2011). Earnings conference calls and stock returns: The incremental informativeness of textual tone. SSRN Working Paper.

Rohal G., Luo Y., Jussa J. and Wang S. (2017). Text mining unstructured corporate filing data. *Wolfe Research Luo's QES*, April 20.

Rohal R., Luo Y. and Jussa J. (2018). Tone at the top? Quantifying management presentation. *Wolfe Research Luo's QES*, January 23.

Rohal R., Luo Y. and Jussa J. (2019). Beyond Fake News. *Wolfe Research Luo's QES*, January 15.

Senter R.J. and Smith, E.A. (1967). Automated Readability Index. Technical Report AMRL-TR-6620. Wright-Patterson Air Force Base.

Sloan R.G. (1996). Do stock prices fully reflect information in accruals and cash flows about future earnings. *The Accounting Review*, 71(3), 289–315.

Sprenger T.O. and Welpe I.M. (2010). Tweets and trades: The information content of stock microblogs, SSRN Working Paper.

Tetlock P., Saar-Tsechansky M. and Macskassy S. (2008). More than words: Quantifying language to measure firms' fundamentals. *Journal of Finance*, 63, (3), 1437–1467.

Wang S., Luo Y., Jussa J. and Rohal G. (2017a). Crowdsourcing Earnings and Revenue Estimates. *Wolfe Research Luo's QES*, June 20.

Wang S., Luo Y., Jussa J., Alvarez M. and Rohal G. (2017b). Machine Learning Takeovers, *Wolfe Research Luo's QES*, September 12.

Zhao F. (2017). Natural Language Processing – Part I: Primer. S&P Capital IQ.

7

Forecasting Beta Using Machine Learning and Equity Sentiment Variables

In this chapter, we apply machine learning, fundamental equity variables and Big Data equity sentiment variables to forecast equity beta. We find that machine learning algorithms are better at forecasting future stock beta than linear models. Big Data variables such as stock level sentiment and news volume are significant in several models in addition to other fundamental variables. The results are statistically significant.

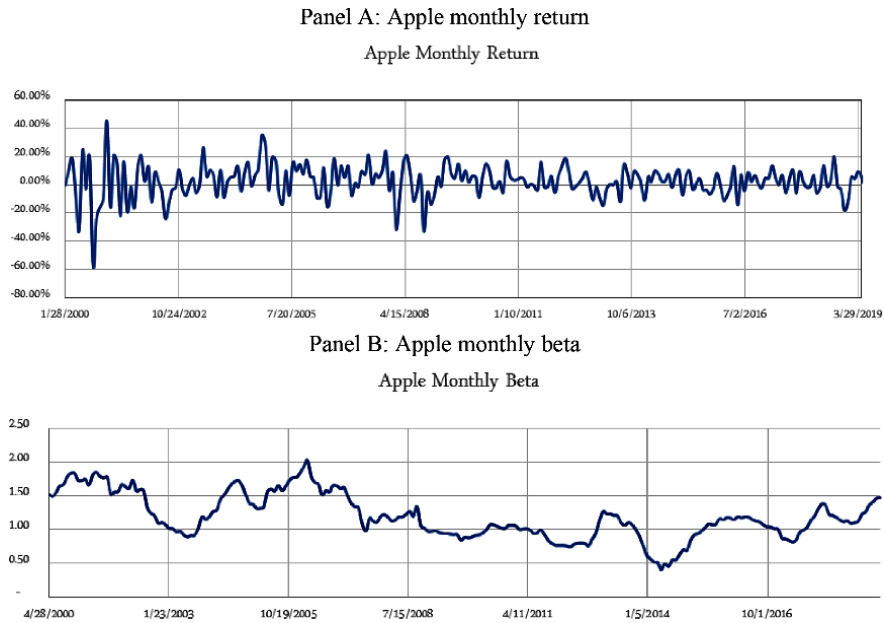
7.1. Introduction

In recent years, there have been several applications of machine learning to the forecasting of expected returns[1]. However, there have been very few studies on the use of machine learning tools to forecast risk characteristics. This is surprising for at least two reasons. First, expected returns are very noisy and difficult to forecast. Figure 7.1 illustrates this in the context of the example of Apple by showing that the monthly mean return of Apple is much more noisy than the monthly beta of Apple. Second, measures of risk such as stock beta have important portfolio management and capital budgeting applications and it would therefore be very interesting to know if machine learning tools can be used to provide superior stock beta forecasts.

Chapter written by Alexei JOUROVSKI, Vladyslav DUBIKOVSKYY, Pere ADELL, Ravi RAMAKRISHNAN and Robert KOSOWSKI.

1 See, for example, Gu *et al.* (2019).

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.



Description: Panel A of this figure shows the monthly average stock return for Apple. Panel B shows the monthly beta of Apple.

Figure 7.1. *Low and high signal-to-noise ratio examples*

In this study, we set out to compare the forecasting ability of standard linear models such as OLS to machine learning algorithms such as LASSO (Least Absolute Shrinkage and Selection Operator), random forest and XGBoost (an implementation of the gradient-boosted decision tree algorithm primarily designed for speed and performance). We use a range of fundamental variables such as valuation ratios, as well as technical indicators such as measures of momentum as the features or predictor variables in the different algorithms. In addition, we also use novel variables – news volume and stock sentiment, provided by RavenPack. These variables are based on natural language processing (NLP) of several terabyte of news stories related to US stocks. Identifying and interpreting the variables that are most important for the forecasting ability of a given algorithm is a challenge. Since the cross-sectional stock features may play a different role in different economic environments, we allow for interaction effects with macroeconomic time-series variables. We use several measures of variable importance, including SHAP (SHapley Additive exPlanations) values. SHAP by Lundberg

and Lee (2016) is a method to explain a feature's average marginal contribution and is based on the game theory optimal Shapley (1953) values.

Our study provides three main findings. First, we show that fundamental variables help improve forecasts of future stock beta. Stock betas are more persistent than expected returns of stocks, and therefore the historical beta is a reasonably good forecast of future beta. We also find that fundamental variables improve the forecasting power of Bloomberg adjusted betas, which are based on shrinkage. News-based variables such as news volume also help to generate forecast improvements in several algorithms. Macro sentiment and interaction terms with fundamental characteristics do not help in a statistically significant way.

Second, we find that tree-based machine learning methods increase the accuracy of beta forecasts in a statistically significant manner compared to simple linear models such as OLS and LASSO regressions. Random forest and XGBoost outperform simpler models in an economically and statistically significant way.

Third, we examine the calculated SHAP values to evaluate the importance of different variables. This allows us to gain insights into the relationship between the feature and the stock beta. We find that many fundamental variables have nonlinear relationships with the forecasted beta values. For example, stocks with both positive and negative momentum tend to lead to an increase in beta, while stocks with neutral momentum do not affect beta forecasts. SHAP analysis can also help detect change in variable importance through time. For example, we find that high values of debt to asset ratio were likely to lead to an increase in beta forecasts prior to 2008, while in the world of low interest rates post 2009, high debt to assets for a stock would imply lower risk.

Our results have important applications for portfolio management, portfolio construction, cost of capital calculations and capital budgeting. Typically, expected returns are based on simple CAPM estimates based on historical beta. Our results suggest that machine learning-based forecasts of beta are superior and could help improve the cost of capital forecasts.

This chapter is related to three streams of the literature. First, it follows the studies that attempt to improve the forecast of beta by means of fundamental variables. Cosemans *et al*. (2016) employ a hybrid approach for

estimating beta that shrinks rolling window estimates towards firm-specific priors motivated by economic theory. Our study differs from the above by using machine learning algorithms to select relevant features including fundamental variables and news-based novel predictors.

Second, we add to the existing literature on the application of machine learning tools to the forecast of expected returns. Gu *et al.* (2019) apply a range of machine learning algorithms including XGBoost to forecast stock returns, while Bianchi *et al.* (2019) forecast expected bond returns. We complement the above work by focusing on the forecasting of stocks' systematic beta risk.

Third, this chapter relates to the literature examining the ability of option implied information and intraday returns to improve beta forecasts. Hooper *et al*. (2008) compare realized measures of beta to constant beta models. Cenesizoglu *et al*. (2017) study beta forecasts for long horizons such as one year. They find that the most accurate forecasts are generated by an autoregressive model of realized beta. Our study is related since we also focus on long horizon beta forecasts, but it also differs in that we apply machine learning algorithms and use fundamental variables to improve beta forecasts.

The rest of this chapter is structured as follows. Section 7.2 describes the data. Section 7.3 discusses the methodology and machine learning algorithms that we apply. Section 7.4 presents and interprets the empirical results. Section 7.5 illustrates how improved beta forecasts could be applied. Section 7.6 concludes.

7.2. Data

7.2.1. *Data construction process*

The sample period for our analysis is 1999–2019. The stock universe is the MSCI US, which contains over time an average of 540 stocks. We exclude financials from our analysis as many fundamental variables that we use are not meaningful for financial or real estate stocks. We use 29 stock characteristics for each stock as explanatory variables or features including historical beta. We also include the interaction of each characteristic with a macroeconomic time series variable and 21 industry group dummy variables according to the

GICS2 classification. This leads to approximately 80 predictors. Table 7.1 shows the list of stock characteristics that we use including their description.

Sub-group	Variable	Description
Momentum	12-month momentum	11-month cumulative returns ending 1 month before month end
Momentum	3-month total return lagged by 1 month	3-month total return lagged by 1 month
Momentum	6-month momentum	5-month cumulative returns ending 1 month before month end
Accounting quality	Accrual ratio	Ratio of the difference between the net income and free cash flow to total assets
Efficiency	Current ratio	Current assets/current liabilities
Efficiency	Quick ratio	(Current assets – inventory)/current liabilities
Efficiency	Interest coverage	Ratio of EBITDA to interest expense
Growth	Sales growth	Annual percent change in sales
Growth	ROE growth	Annual percent change in ROE
Growth	EPS growth	Annual percent change in earnings per share
Investment	Asset growth	Annual percent change in total assets
Profitability	Return on equity	Ratio of income before extraordinary items to common shareholders' equity
Profitability	Earnings-per-share	Ratio of income before extraordinary items to shares outstanding
Profitability	Return on invested capital	EBIT minus income taxes scaled by invested capital
Profitability	Operating profitability	Ratio of EBIT to revenue
Profitability	Return on assets	Ratio of income before extraordinary items to total assets
Profitability	OI after depreciation to operating assets	Ratio of operating income after depreciation to operating activities
Safety/solvency	Debt/assets	Ratio of long-term debt to assets
Safety/solvency	Net debt to EBITDA	Ratio net LT debt (total LT debt – cash equivalents) to EBITDA

Safety/solvency	Asset leverage	Ratio of assets to stockholders equity
Risk	Return volatility	Standard deviation of daily returns from month t-1
Risk	Beta	Historical beta from month t-1
Sentiment	Sentiment average	Sentiment average
Sentiment/size	Monthly volume	Monthly volume
Size	Market capitalization	Natural log of market capitalization at the end of month t-1
Value	Book-to-market	Ratio of book value to market capitalization
Value	Earnings-to-price	Ratio of income before extraordinary items divided by market cap
Value	Sales-to-price	Ratio of sales to market cap
Value	EBITDA/enterprise value	Ratio of EBITDA to enterprise value

Description: This table describes the features used in this study and is grouped by sub-category.

Table 7.1. *Variable list*

The characteristics can be grouped into several sub-groups which include price-based technical indicators such as momentum signals, quality indicators such as different measures of profitability, market capitalization, equity level sentiment, risk measures such as historical beta and volatility, as well as measures of valuation such as book-to-market and earning-to-price measures. The motivation for these measures is based on both academic literature and industry practice of fundamental analysts that assess the risk of stocks. Two relatively novel measures of risk are the sentiment-based measures from RavenPack, namely the average stock sentiment and the monthly news volume.

The stock-level sentiment measure is based on the Event Sentiment Score (ESS) developed by RavenPack. ESS depends on detecting an event category and other information that is extracted from a news story such as financial figures, analyst ratings and directional or emotional language. It can be argued that ESS takes context into account and hence should provide a precise sentiment score. For more information about the ESS, we refer the reader to Hafez and Xie (2016).

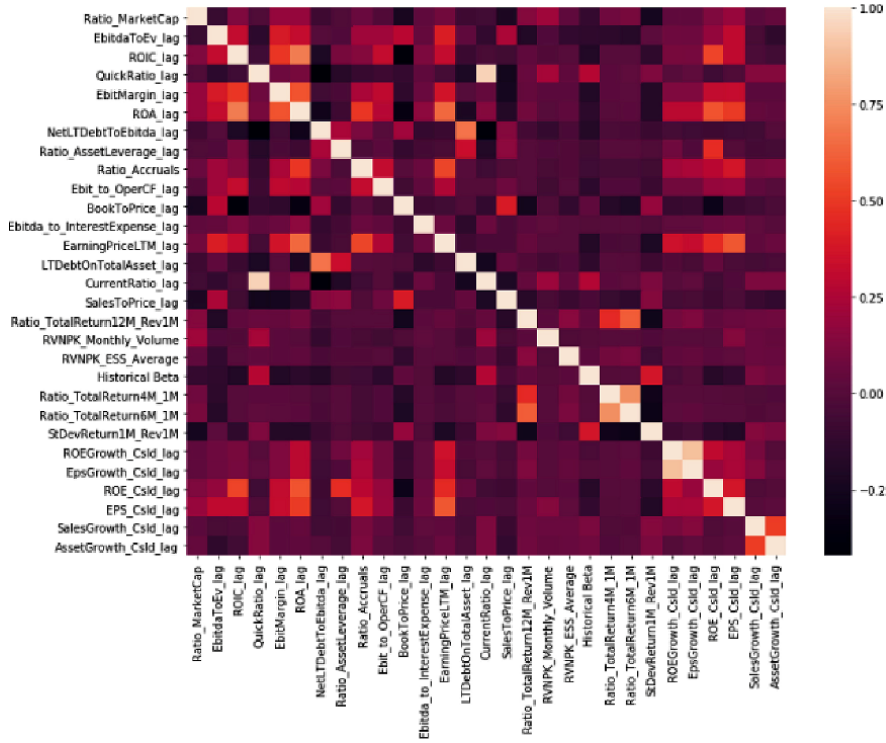
As part of our data cleaning process, we have winsorized the data. Table 7.2 shows the summary statistics for the winsorized data.

Fundamental variable	Mean	Median	Min	Max
TotalReturn12M	0.123969	0.106976	–0.820073	1.607227
TotalReturn4M	0.030697	0.032656	–0.538713	0.609319
TotalReturn6M	0.052275	0.052456	–0.644967	0.82891
AccrualsRatio	–0.003036	-0.002648	–0.414226	0.255181
CurrentRatio	1.845781	1.49689	0.303089	9.548755
QuickRatio	1.460363	1.119996	0.169955	9.158574
Ebitda_to_IntExp	20.08076	7.79553	–7.811111	731.9483
SalesGrowth	1.097713	1.065882	0.507392	2.729407
ROEGrowth	1.041891	1.002127	–11.769094	15.920623
EpsGrowth	1.16537	1.102681	–7.596366	14.627429
AssetGrowth	1.125512	1.059421	0.574551	3.770861
ROE	0.178173	0.15282	–1.668978	3.167758
EPS	0.000002	0.000002	–0.000021	0.000031
ROIC	0.137094	0.123363	–0.371943	0.751946
EbitMargin	0.146614	0.144125	–1.212449	0.550923
ROA	0.062425	0.06132	–0.44712	0.310457
Ebit_to_OperCF	1.010035	0.977055	–2.868088	6.605809
LTDebtOnTotAsset	0.228715	0.214507	0	0.915139
NetLTDebtToEbitda	0.975524	0.879338	–11.34353	10.3065
Ratio_AssetLeverage	3.424003	2.453093	1.113257	49.65281
StDevReturn1M	0.019181	0.015922	0.004564	0.086002
HistoricalBeta	1.036279	0.998597	–0.939692	4.817595
Sentimentavg.	58.92109	60	0	100
MonthlyVolume	1545.809	291	1	486523
MarketCap	4.123675	4.051532	3.07083	5.521012
BookToPrice	0.41145	0.333411	–0.215436	2.644082
EarningPriceLTM	0.039443	0.048958	–0.879406	0.273845
SalesToPrice	0.934538	0.588987	0.036411	8.815965
EbitdaToEv	0.080435	0.077572	–0.10132	0.248498

Description: This table shows the summary statistics of the winsorized data.

Table 7.2. *Summary statistics*

To illustrate the correlation between features used in this study, Figure 7.2 shows the heat map of the Pearson correlation matrix. We observe strong positive correlations ($\rho > 0.8$) between the Current and Quick ratios ($\rho = 0.947$) and the earnings per share (EPS) and return on equity (ROE) growth ($\rho = 0.891$).

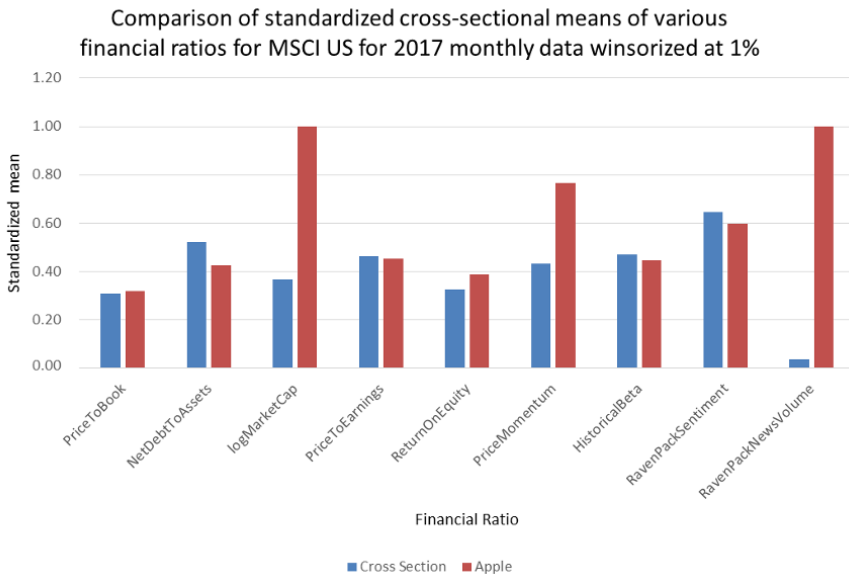


Description: This figure shows a heat map of the Pearson correlation matrix between the variables used in this study.

Figure 7.2. *Heat map of the Pearson correlation matrix. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

To further illustrate the fundamental variables used, Figure 7.3 shows how the values of some financial ratios for Apple in our sample compare to the cross-sectional means. We standardize the variables for presentation in this figure so that the minimum value corresponds to 0 and the maximum value corresponds to 1. For news volume, the Apple stock (red bar) has a value of 1 which means that Apple has the highest value of monthly news

volume. The cross-sectional mean is below 5%, which implies that the average company has a relatively small number of news stories per month.



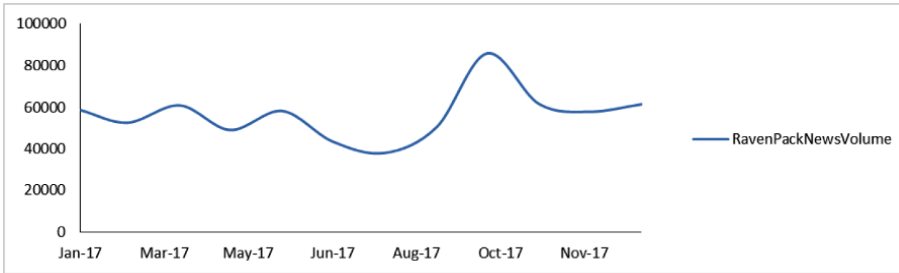
Description: This graph compares the average of various fundamental variables with the value of those variables for the stock of Apple. The values are standardized so that the variables are comparable. The standardization of the (average minimum)/(maximum minimum). The time series began in January 2000 and ended in December 2018. The stock universe is that of the US Russell 1500. Source: RavenPack, Unigestion.

Figure 7.3. *Big Data example – Apple: RavenPack equity news volume and sentiment. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

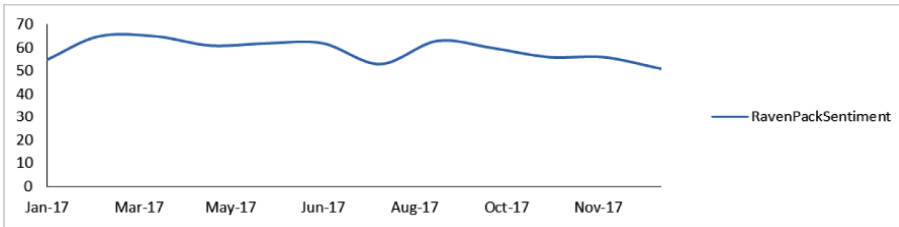
The news volume and stock sentiment measures are based on natural language processing and Big Data application. The measures are relatively recent, and some readers may not be familiar with them. Hence, we illustrate the measures by means of the example of Apple in Figure 7.4. Panel A shows the number of news stories per month over the course of 2017 for Apple. There is a peak in news volume for Apple around September 2017. This can be explained by the release of three new iPhone models on September 12, 2017, which led to an unusually large amount of news volume around the release date. Panel B of Figure 7.4 shows the news sentiment that is the average mood of news articles on the subject of Apple. A value above (below) 50 indicates a positive (negative) mood. In

2017, the news sentiment measure for Apple increased before and declined after September 2017.

Panel A: Number of news stories



Panel B: Sentiment score



Description: The top panel shows the number of monthly news articles about Apple. The time period is from January 2017 to December 2017. The bottom panel shows the news sentiment for Apple, which is the average mood of news articles on the subject of Apple. A value above (below) 50 indicates a positive (negative) mood. Source: RavenPack, Unigestion.

Figure 7.4. *Big Data example – Apple: RavenPack equity news volume and sentiment*

7.3. Methodology

We apply six different methodologies to forecast stock beta. These include (a) the simple historical beta, (b) the Bloomberg adjusted beta, (c) a linear forecast based on an OLS regression, (d) a LASSO regression, (e) a random forest model and (f) an XGBoost model. In the following, we briefly describe the methodologies and refer to publications that describe these methods in further detail.

7.3.1. *Historical beta*

The simplest forecast that we use for a stock's beta is its historical beta, which is calculated at the end of each month based on the daily returns during the year leading up to and including this month.

7.3.2. *Bloomberg's adjusted beta*

It was observed by Vasicek (1973) that historical beta underestimates future beta for low risk stocks and overestimates future beta for high risk stocks. Therefore, an improved forecast of future beta can be obtained by shrinking the historical beta towards one. From a Bayesian point of view, we can view the adjusted beta as imposing certain prior distributions on model parameters. This is how Bloomberg calculates the adjusted beta. The formula used by Bloomberg is adjusted beta = (0.67) × historical beta + (0.33) × 1.0.

7.3.3. *OLS regression*

The simplest regression methodology that we apply is a linear OLS regression of beta on lagged fundamental variables described in Table 7.2.

7.3.4. *Post-LASSO OLS regression*

In order to improve the simple OLS method, we also apply a LASSO (Least Absolute Shrinkage and Selection Operator). LASSO has the advantage of performing both variable selection and regularization, which can enhance the prediction accuracy and reduce overfitting. The tuning hyperparameter, λ, controls the strength of the L1 penalty, which can be interpreted as the amount of shrinkage used. This parameter is selected by minimizing the MSE (mean squared error) of the forecast in the validation dataset. The reason why the LASSO regression can be used to reduce overfitting and for model selection is that the L1 penalty often leads to zero coefficients, i.e. some of the features are completely dropped for the evaluation of the output. As is a standard practice with using LASSO, we rerun OLS following a LASSO step while excluding variables that were identified by LASSO as irrelevant at the optimal level of L1 penalty.

7.3.5. *Random forest model*

The random forest model is an ensemble classification algorithm based on a modified tree-learning algorithm. It includes a feature-bagging step that is based on a random selection of subsets of features. The idea here is to train the various learners of the ensemble on different feature sets in parallel to reduce the correlation between them and increase the accuracy of the ensemble algorithm.

7.3.6. *XGBoost model*

XGBoost (Extreme Gradient Boosting) is an implementation of the gradient boosting algorithm. Similar to random forest, gradient boosting is a decision tree-based ensemble machine learning algorithm. Following this method, sequential forecasting is performed where subsequent predictors learn from the mistakes of the previous predictors. The observations thus have an unequal probability of appearing in subsequent models, and the ones with the highest error in the previous models appear most in the following ones. This is what makes this algorithm different from random forest where observations are chosen based on a bootstrap process and not on the forecasting error of the previous models. The trade-off of this algorithm against random forest is that while both reduce model variance due to their ensemble nature, gradient boosting also reduces bias but at the cost of potentially overfitting.

7.4. Empirical results

7.4.1. *Variable selection*

In a preliminary step, we apply the LASSO regression model for the purpose of variable selection. Table 7.3 compares the OLS with the LASSO regression results. Several interesting insights are obtained from the table. As column 7 shows the LASSO, regression identifies the historical beta as the most relevant predictor variable. Larger stocks have a lower forecast beta (Ratio_MarketCap). Stocks with positive momentum (Ratio_TotalRetrun 12M_Rev1M) exhibit lower predicted beta. The RavenPack variables are not significant. The dummies (Food & Beverage, Utilities) show that Food and Beverage and Utility stocks have lower beta than the average stock.

Coef	OLS	t	P>|t|	LASSO	t	P>|t|
const	1.037	1172.22	0	1.0376	1189.86	0
Ratio_MarketCap	–0.0011	–1.06	0.289	–0.0036	–3.869	0
EbitdaToEv_lag	–0.0043	–3.291	0.001			
ROIC_lag	0.0035	2.324	0.02	–0.0047	–3.968	0
QuickRatio_lag	0.004	1.147	0.252	0.0173	6.156	0
EbitMargin_lag	–0.0019	–1.508	0.132	–0.0059	–5.57	0
ROA_lag	–0.0204	–9.192	0			
NetLTDebtToEbitda_lag	–0.0128	–8.96	0			
Ratio_AssetLeverage_lag	0.0002	0.147	0.883			
Ratio_Accruals	0.0213	15.309	0			
Ebit_to_OperCF_lag	–0.0017	–1.65	0.099			
BookToPrice_lag	0.0198	15.245	0	0.0159	13.77	0
Ebitda_to_InterestExpense_lag	0.0006	0.766	0.443			
EarningPriceLTM_lag	–0.0062	–3.906	0			
LTDebtOnTotalAsset_lag	0.0089	7.078	0			
CurrentRatio_lag	0.0129	3.771	0	–0.0002	–0.056	0.955
SalesToPrice_lag	0.0108	8.808	0	0.0098	8.409	0
Ratio_TotalReturn12M_Rev1M	0.038	31.74	0	0.0374	31.584	0
RVNPK_Monthly_Volume	–0.0012	–3.557	0			
RVNPK_ESS_Average	–0.003	–3.432	0.001			
HistoricalBeta	0.2216	189.089	0	0.2258	194.998	0
Ratio_TotalReturn4M_1M	0.0043	3.053	0.002			
Ratio_TotalReturn6M_1M	0.0124	7.843	0	0.016	13.455	0
StDevReturn1M_Rev1M	0.0728	64.015	0	0.0737	66.02	0
ROEGrowth_Csld_lag	–0.0088	–4.205	0			
EpsGrowth_Csld_lag	0.0104	5.021	0			
ROE_Csld_lag	–0.0009	–0.687	0.492	–0.0031	–3.209	0.001
EPS_Csld_lag	–0.0007	–0.702	0.483			
SalesGrowth_Csld_lag	0.0071	6.559	0	0.0075	7.047	0
Energy	0.0235	26.93	0	0.0203	21.593	0
Materials	0.0138	16.964	0	0.0088	9.709	0
CapGoods	0.0199	25.013	0	0.0131	14.159	0
CommercServices	0.0008	0.881	0.378			
Transport	0.0053	6.658	0			
Auto&Compon	0.0103	12.416	0	0.0082	9.614	0
ConsumerDurables	0.008	9.713	0			
ConsumerServices	0.0069	8.129	0			
Media	–0.0047	–5.414	0	–0.0099	–10.403	0
Retailing	0.0009	0.927	0.354			
StaplesRetailing	–0.0191	–20.813	0	–0.0205	–21.76	0
Food&Bev	–0.0325	–38.733	0	–0.0357	–39.039	0
HouseholdProd	–0.0152	–17.854	0	–0.0174	–19.759	0
HealthEquip&Serv	–0.0229	–27.966	0	–0.0288	–30.773	0

Coef	OLS	t	P>|t|	LASSO	t	P>|t|
Pharma&Biotech	–0.0083	–10.074	0	–0.0128	–14.221	0
Software	0.0137	15.95	0			
ITHardware	0.022	24.479	0	0.0154	15.602	0
Semicond	0.0194	21.456	0	0.0141	14.926	0
Telecom	–0.0061	–6.682	0			
Utilities	–0.0462	–49.01	0	–0.0492	–47.942	0
AssetGrowth_Csld_lag	0.0051	4.998	0	0.0054	5.371	0

Description: This table reports LASSO regression results for the purpose of variable selection. Columns 2–4 report the OLS, and columns 5–7 report the LASSO regression results.

Table 7.3. *LASSO*

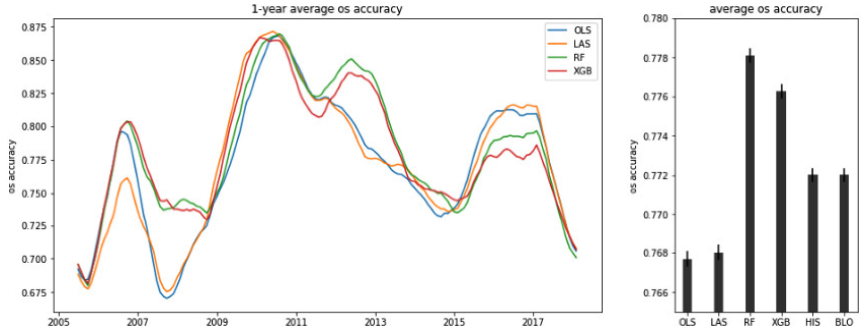
The Post-LASSO OLS regression results are full-sample results. Since our objective is to allow different algorithms to employ the whole range of features at different points in time, we do not restrict the list of variables to the ones that are identified as significant in Table 7.3.

7.4.2. *Forecasting models*

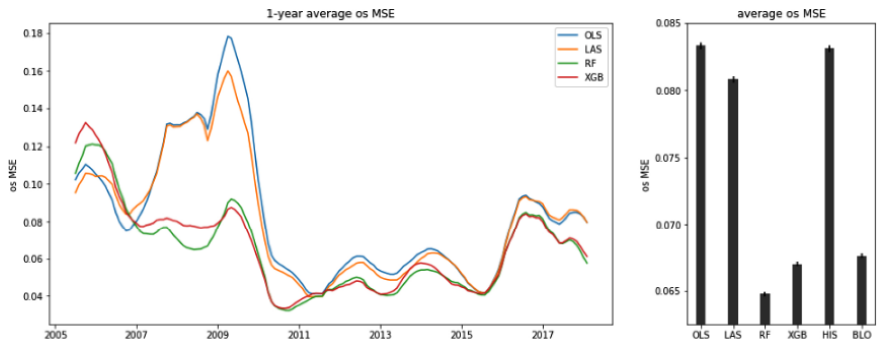
In our main empirical application, we use forecasts from the different models to forecast stocks' beta. We report three measures of the models' performance, which are closely related. First, we show the out-of-sample accuracy which is defined as the percentage of correctly identifying whether one-year realized future beta will be above or below a cutoff value (in hindsight). We use the cutoff value of 1.0 and evaluate how often we are correct in determining whether the systematic (CAPM) risk of a stock is below or above that of the market. Second, we report the MSE, which is the average squared difference between the realized future beta and the forecast beta. Third, we report the cross-sectional correlation between the forecast beta and the realized beta. The square value of this correlation gives the out-of-sample R-squared of the forecast.

Figure 7.5 plots the one-year average out-of-sample accuracy over time and shows a bar chart with the full-time average out-of-sample accuracy. The bar chart shows that the OLS and LASSO models underperform the Bloomberg and historical beta estimates. However, it is the random forest algorithm that outperforms all the other methods. There is some time-variation in the relative performance of the different models. The random forest model performs well on average but experiences a period of underperformance around 2016 when the LASSO model is superior in terms of accuracy.

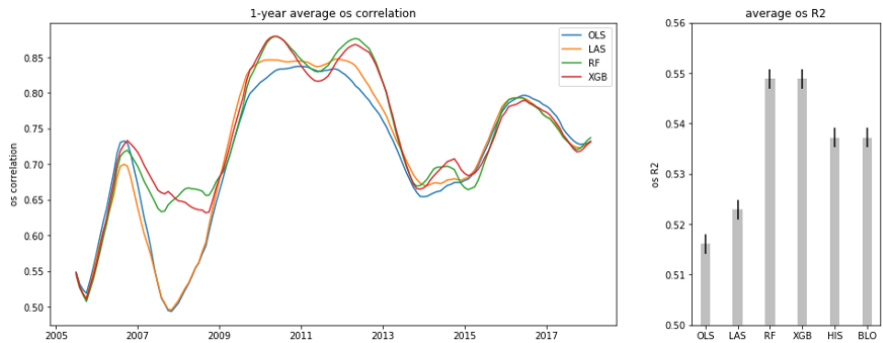
Panel A: Accuracy



Panel B: MSE



Panel C: Correlation



Description: Panel A shows the out-of-sample accuracy which is defined as the percentage of correctly identifying whether the actual future beta will be above or below a cut-off value. In Panel B, we report the MSE (mean squared error), which is the average squared difference between the actual beta and the forecast beta. Panel C reports the cross-sectional correlation between the forecast beta and actual beta.

Figure 7.5. *Comparison of forecasting methods. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

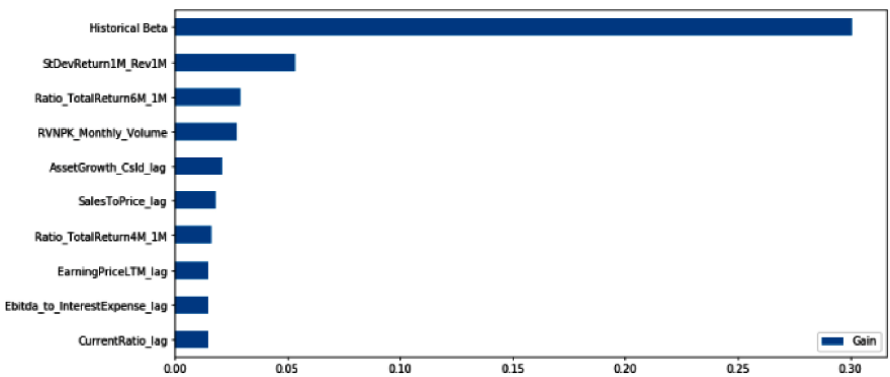
Panel B of Figure 7.5 shows the average out-of-sample correlation and the one-year average out-of-sample correlation over time. XGBoost is found to be superior in terms of out-of-sample R-squared.

Panel C of Figure 7.5 shows the MSE. On average, the random forest model generates the lowest MSE, closely followed by XGBoost. We conclude from the above analysis that the random forest model generates the most accurate forecasts.

7.4.3. *Variable importance*

In this section, we discuss the importance of different variables used by the models discussed above and how these variables impact model performance. To do this, we use two interpretation techniques. The first one is a standard interpretation method called gain. The second method is the SHAP value, a method that has gained popularity over the last two years and the only method consistent with game-theoretical considerations. We follow the recent application of SHAP values to explain the XGBoost forecasting performance in Chew *et al*. (2019).

Gain is defined as the total reduction in the loss function accounting for all splits for a given feature. A feature that produces a decrease in loss throughout the tree is considered relevant. Figure 7.6 plots the average gain feature importance of XGBoost. We see that the measures of historical risk such as historical beta and standard deviation dominate.



Description: This plot shows the top 10 average gain feature importance of XGBoost for each prediction.

Figure 7.6. *Feature importance for XGBoost*

Figure 7.6 shows the average of the gain feature importance when the XGBoost model is used for predictions. We observe that the historical beta's gain is approximately five times bigger in magnitude than any other feature's gain.

7.4.4. *SHAP values*

Lundberg and Lee (2017) propose an innovative interpretation model called SHapleyAdditive exPlanation (SHAP) values that theoretically allow us to interpret the machine learning predictions in a reliable and robust way. We will explain how it works and further apply this technique to our XGBoost predictions of beta, whose extension efficiently runs on tree ensembles. We obtain SHAP values for each prediction throughout time and propose different visualization frameworks providing better insights on how the algorithm forecasts beta.

To understand Shapley values, assume a game setup with N players, where S is a subset of the N players. Let $v(S)$ be the value these S players bring to the game. When player i joins the S players, player i's marginal contribution is $v(S \cup \{i\}) - v(S)$. If we take the average of the contribution over the different possible permutations in which the coalition can be formed, we get the right contribution of player i. The Shapley value for feature i is thus defined as:

$$\boldsymbol{\varphi_i(v)} = \sum_{\boldsymbol{S \subseteq N \setminus \{i\}}} \frac{\boldsymbol{|S|!\,(N-|S|-1)!}}{\boldsymbol{N!}} \boldsymbol{(v(S \cup \{i\}) - v(S))} \qquad [7.1]$$

The intuition behind equation [7.1] is that the Shapley value for a feature is the difference in prediction with and without that feature in the model. The reason for using a combination is that the order in which a model assesses features can affect its predictions, i.e. at what point in the tree we use the feature to split the tree.

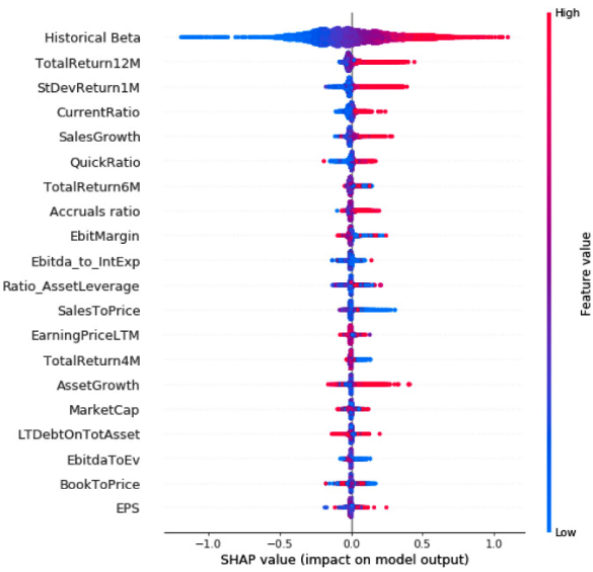
Thanks to the recent contribution of Lundberg and Lee (2017), it is possible to approximate the Shapley values and compute them with a feasible computational complexity. Based on these insights, Lundberg *et al.* (2018) introduced a tree-based method that runs in $O(TLD^2)$ time, where T is the number of trees, L is the maximum number of leaves in any tree and D is the maximal depth of any tree. In practice, this is done by recursively keeping track of the proportion of all possible subsets that flow down into each of the tree leaves.

We use the SHAP Python library to compute the SHAP values. Since SHAP is a local method, we compute the importance of each feature at making individual predictions. We obtain one SHAP value per feature per observation. As we use an expanding window, the number of SHAP values per prediction increases over time.

For the sake of interpretability of our plots, a SHAP value of one represents a 100% impact on model output.

7.4.5. *Overall level of feature importance*

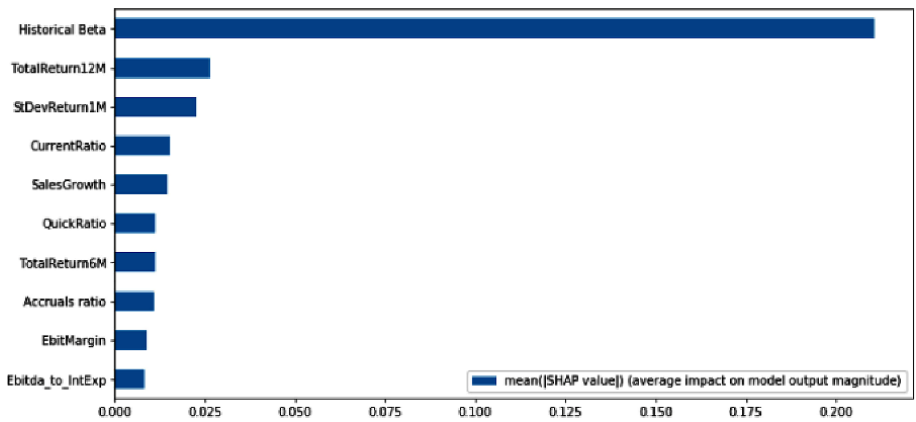
Figure 7.7 provides a visual representation of the importance values for each feature. We select the top 20 features and sort them by a SHAP feature importance magnitude. The figure shows that beta has a positive impact when it is high (red dots) and a negative impact when it is low (blue dots), implying that the model correctly captures a linear relationship between historical and future realized beta.



Description: This figure provides a visual representation of the importance values for each feature. Each feature is associated with a number of dots equal to the number of stocks predicted, and each dot is the impact (x-axis) of the feature on the prediction of the individual stock. The color of the stock corresponds to the value of the feature.

Figure 7.7. *SHAP values by feature. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Looking at the first five features, we can see that the algorithm is choosing stocks with high 12-month momentum (TotalReturn12M), high exposure to the market (historical beta), high standard deviation (StdDevReturn1M), high current ratio and high sales growth. For the case of sales-to-price, low sales values contribute to an increase in the beta forecast. Long-term debt on total assets only has positive and negative SHAP values for high values of the features, indicating that the algorithm is not taking a position either in favor or against them on average. From a feature importance comparison's point of view, we are more interested in the level of impact rather than the sign. Hence, the average absolute impact of each feature is computed, and we report the top 10 features in Figure 7.8. Taking the absolute value allows the consideration of both negative and positive impacts of each feature. Comparing it to Figure 7.6, which is based on the gain method, there are clear differences. For instance, RVNPK monthly volume, in the top 4 in the gain importance, does not even appear on the top 20 for the SHAP values. The momentum features appear to be more important when using the SHAP values. SHAP replaces the asset growth by the sales growth. Finally, historical beta still accounts for a majority of the explanatory power and standard deviation still remains important, although its contribution is somewhat reduced.



Description: Feature importance – average absolute SHAP value.

Figure 7.8. *SHAP values by feature*

7.4.6. *Cross-sectional analysis of feature importance*

In order to gain insight into how individual feature values affect forecasted beta, we focus on cross-sectional analysis first.

This analysis enables us to spot nonlinear relationships identified and used by the XGBoost algorithm in beta forecasting.

First, let us look at the most important variable – historical beta in Figure 7.9.

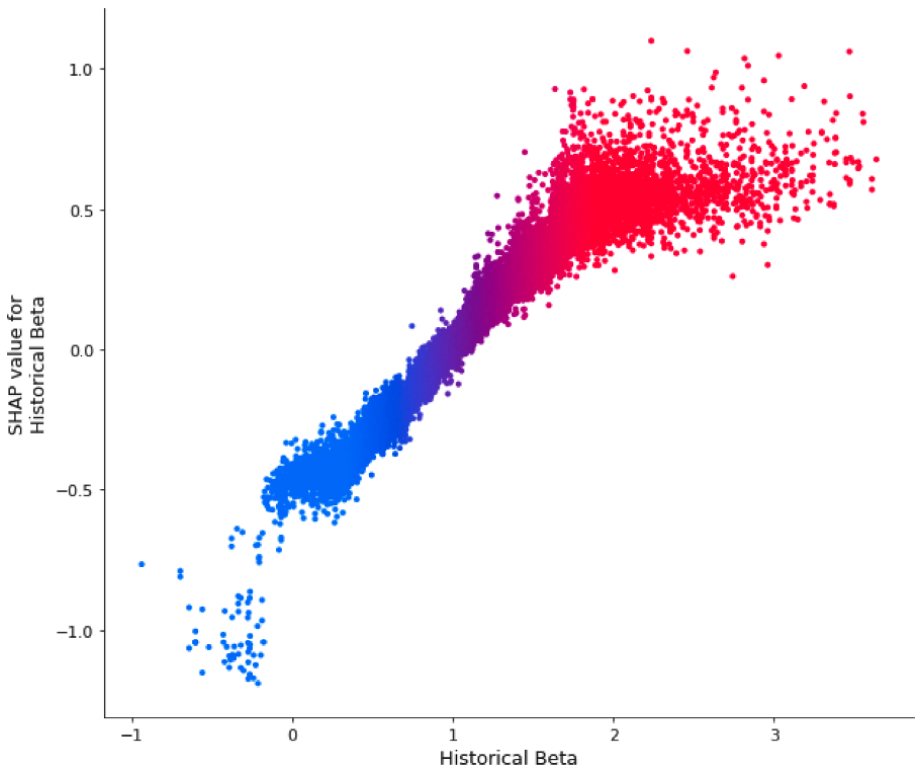
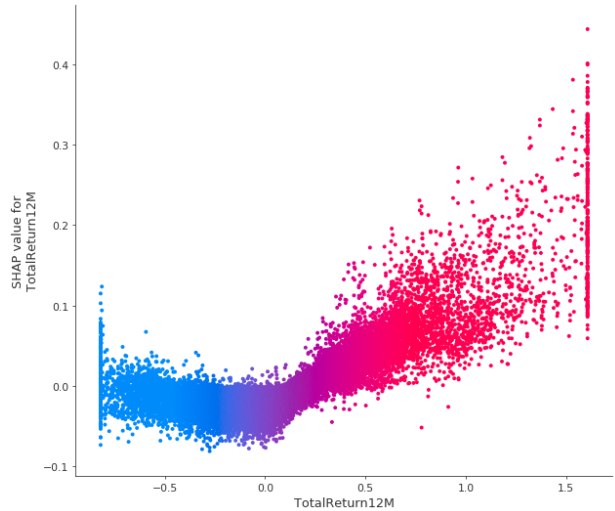


Figure 7.9. *Dependence plot of the SHAP values on historical beta, colored by a historical beta value. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

In Figure 7.9, each dot refers to one stock. The color on the plot represents the distribution of the feature values, i.e. purple is the middle region and red and blue are the extremes. In this case, the purple points have a beta slightly higher than one, the red points have very high betas and the blue ones have a beta close to zero. There is an expected linear relationship – high values of historical beta lead to higher SHAP values (increase in beta forecast compared to average), and lower values of historical beta lead to negative SHAP values (decrease of beta forecast compared to average). What is more interesting, is that we see some nonlinear effects in this plot. We observe that the historical beta values higher than 2 do not add importance following the previous effect but remain on the same level of importance and start to disperse vertically. The higher dispersion of this subset of stocks indicates that there are potentially other features that contribute to the importance – a potential interaction effect. On the negative extreme, there is a jump for negative historical beta values that affect the output with SHAP values of approximately -1.

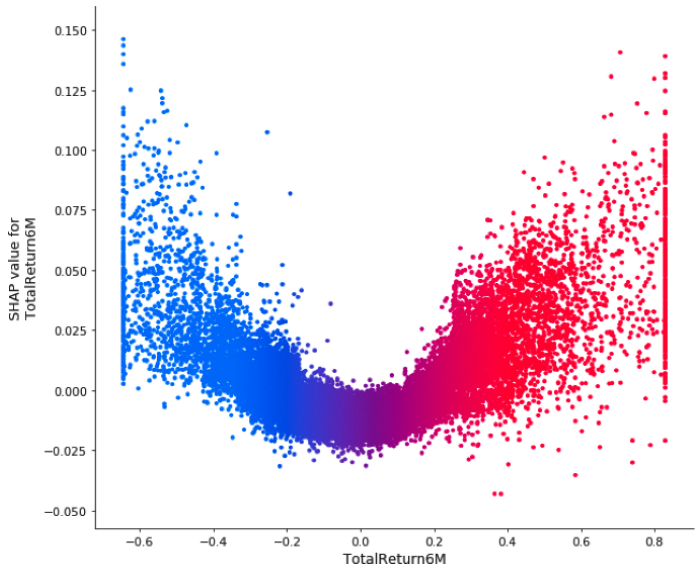
Figure 7.10 shows the importance of 12-month momentum. We observe that unlike in the previous graph, here, there is only a positive impact or no impact. The dependence is strongly nonlinear. The 12-month momentum feature gains importance as positive returns increase – i.e. companies that have been performing well during the last year are likely becoming riskier – this could be indicative of momentum crashes. Companies that have small positive or negative returns have no impact on assessing that stock's beta.

Examining the 6-month momentum in Figure 7.11, we observe that a negative return over that period does increase the forecast of beta for that stock, and this trend increases with higher dispersion as returns get more negative. Positive returns affect the beta forecast in the same manner as before but the magnitude is smaller and now similar to the negative tail. This suggests that as the momentum window size decreases, negative returns gain importance in beta forecasting and positive returns decrease in importance.



Description: This figure shows the SHAP values for the 12-month momentum feature.

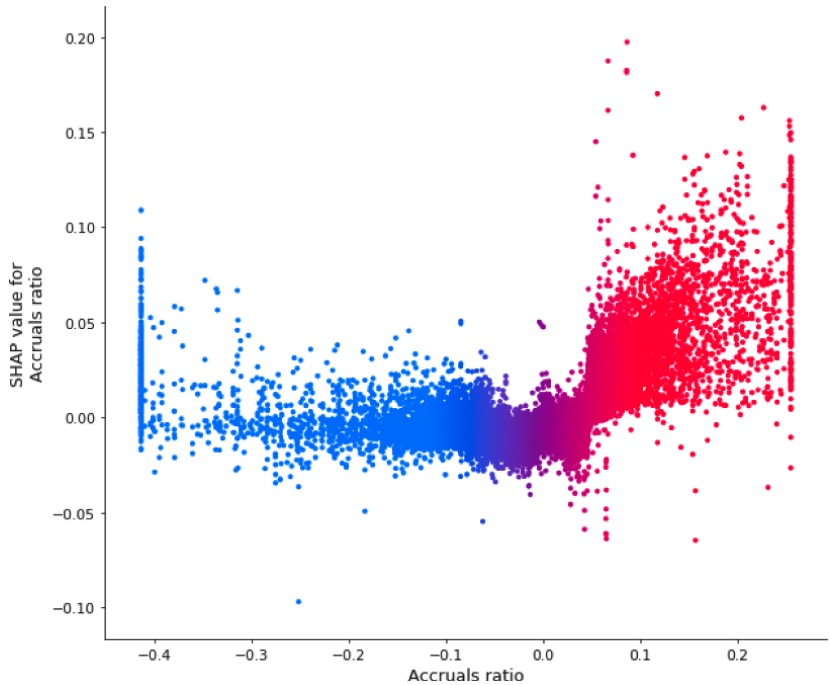
Figure 7.10. *Dependence plot of the SHAP values on 12-month momentum. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*



Description: This figure shows the SHAP values for the six-month momentum feature.

Figure 7.11. *Dependence plot of the SHAP values on six-month momentum. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Finally, Figure 7.12 shows some nonlinearities found in the quality-type fundamental variable accrual ratio. Negative and small positive accruals do not affect the value of the beta forecast. However, when accruals are high, there is a jump-like effect on the forecasted beta (with increased dispersion).



Description: This figure shows the SHAP values for the accrual ratio feature.

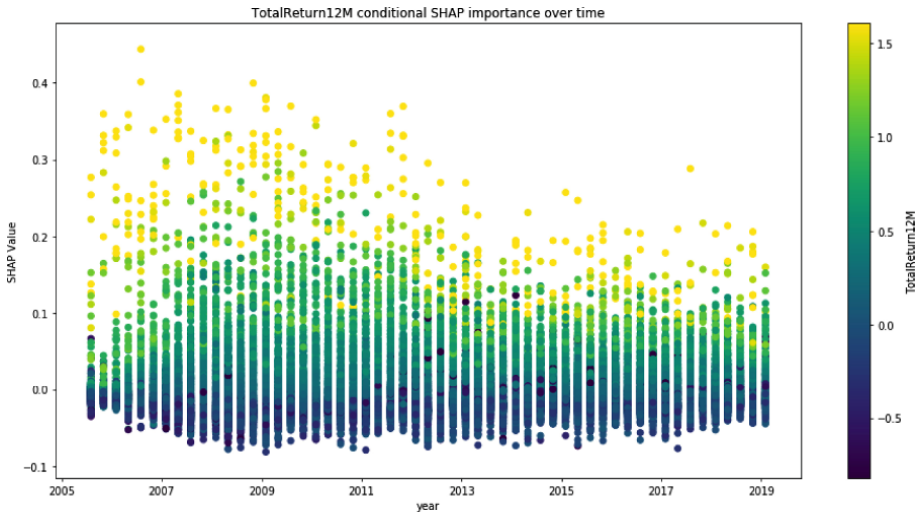
Figure 7.12. *Dependence plot of the SHAP values on accrual ratio. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

7.4.7. *Time-series analysis of feature importance*

The general problem in the analysis in the previous section is that we plot the distribution of the observations from all the predictions, and therefore we do not account for changes in time. Seeing the distribution through time may help understand, for instance, some of the non-intuitive results that show both positive and negative SHAP values for the same range of feature values. This section focuses on feature values that drive the beta forecast

over time (from 2005 to 2019). We focus on how the feature values change both through time and through the addition of more data, given that at every training step, we expand the window size.

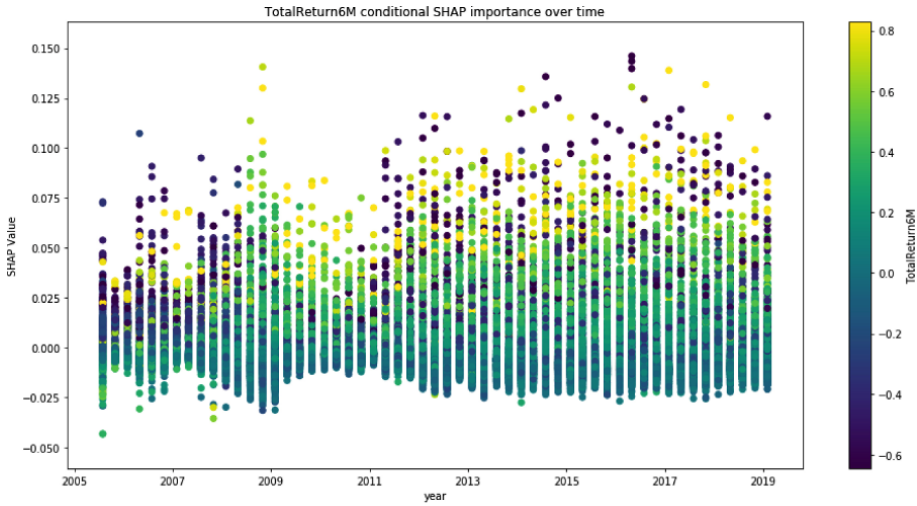
We now focus on the momentum features. Figure 7.13 shows the 12-month momentum with high returns being important between 2005 and 2012. In addition, the medium positive returns start lower but gradually become more important. Post 2013, dispersion decreases and positive returns lose importance. If we go to lower window sizes for momentum, we can observe the opposite pattern in time. This suggests that, as more data is fed, the algorithm is able to understand better smaller frames – thus the algorithm is able to improve the relationships with those signals.



Description: This figure shows the historical evolution of the 12-month momentum feature SHAP value.

Figure 7.13. *Historical evolution of the 12-month momentum feature SHAP value. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Figure 7.14 shows the opposite effect for six-month momentum – it becomes more important after 2012, with a spike in 2009. It is interesting to observe how the negative returns positively affect beta: at first, they are concentrated but not so dispersed with positive but small SHAP values but post the great recession, they lose importance and then gain it again in mid-2011.

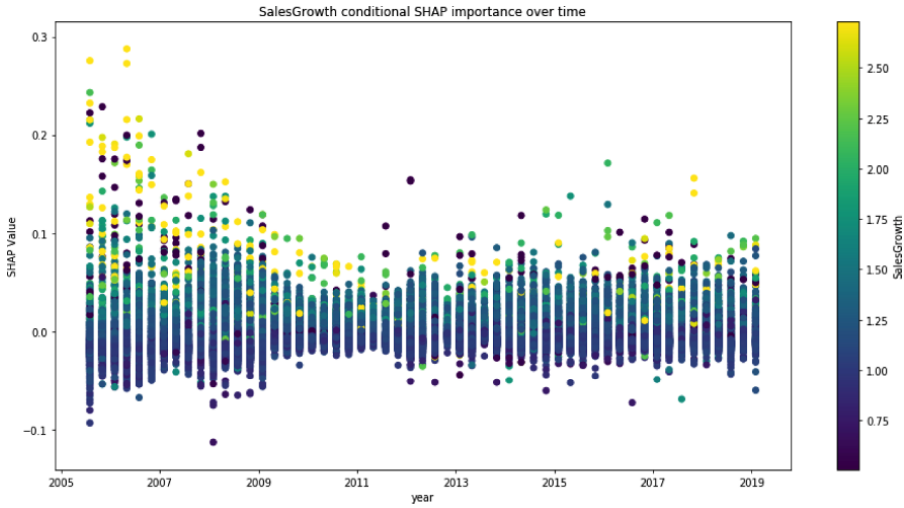


Description: This figure shows the historical evolution of the six-month momentum feature SHAP value.

Figure 7.14. *Historical evolution of the six-month momentum feature SHAP values, colored by the feature value. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

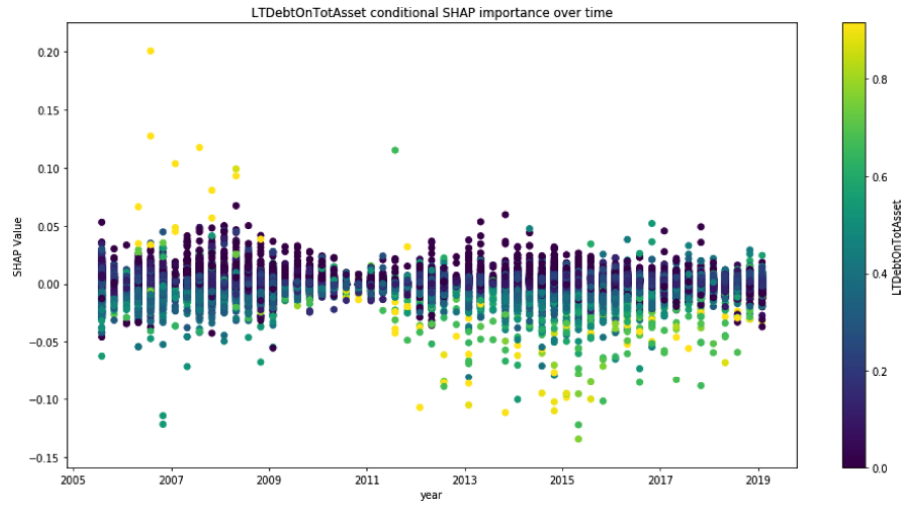
The last set of variables that we illustrate are sales growth and long-term debt to total assets. The reason we have selected these variables is because they seem to change in time, and we would like to try to understand the economic insight behind it. In Figure 7.15, sales growth has a strong positive impact for both high and low values in the early years (2005–2009) and after that, they become less relevant. The fact that high values of sales growth alter the beta forecast in a positive manner is indicative of the inherent riskiness of high-growth stocks.

Finally, in Figure 7.16, high long-term debt on total assets drives beta forecast up until 2009 and after that, it drives it down. This can be interpreted as owning debt being risky before the great financial crisis, then, as quantitative easing begins and interest rates drop – piling up debt seems to indicate a reduction in the risk profile of a company.



Description: This figure shows the historical evolution of the sales growth feature SHAP values .

Figure 7.15. *Historical evolution of the sales growth feature SHAP values. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*



Description: This figure shows the historical evolution of the long-term debt on the total asset feature SHAP values.

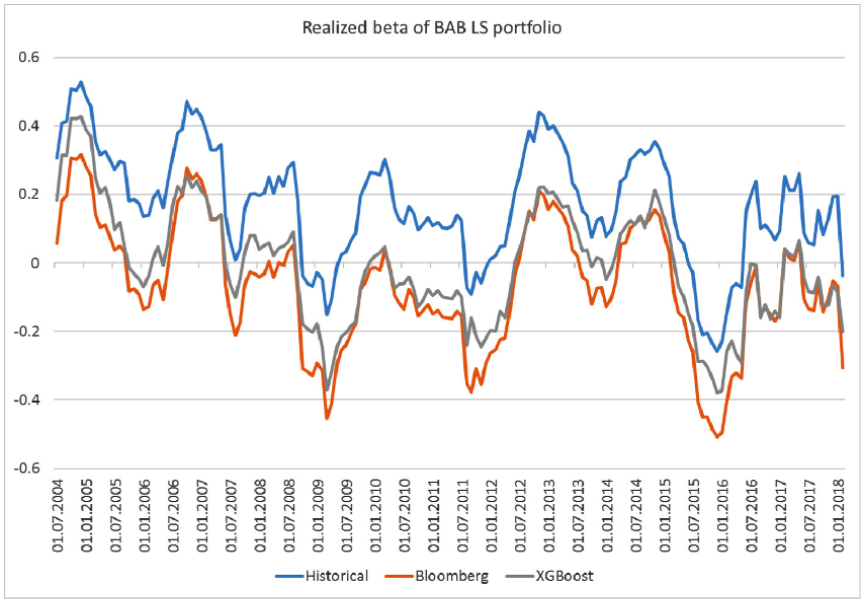
Figure 7.16. *Historical evolution of the long-term debt on the total asset feature SHAP values. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

7.5. Constructing market neutral long–short portfolios

To illustrate how improved beta forecasts can be used in practice, we construct market neutral long–short portfolios based on the Betting-Against-Beta factor reported by Frazzini and Pedersen (2014). We sort stocks in the MSCI USA universe (excluding financials and real estate sectors) according to their historical betas. We go long the decile of stocks with lowest historical beta (D1) and go short the stocks in the highest historical beta decile (D10). Both long and short portions are equally weighted (EW). We adjust the short portion of the portfolio to maintain market neutrality of the long–short combination using three different methods: 1) ratio of historical betas of EW D1 and EW D10; 2) ratio of Bloomberg betas of EW D1 and EW D10; and 3) ratio of XGBoost beta forecasts of EW D1 and EW D10. We evaluate whether realized 12-month betas of the three long–short portfolios are different from zero. We repeat the procedure each month and calculate significance using the Newey–West t-statistics to account for autocorrelation.

Figure 17.1.7 shows the realized betas for all three long–short portfolios. As expected, when sizing the short sub-portfolio using the ratio of historical betas, we get significantly positive *ex post* beta for the long–short portfolio with a t-statistic of 5.5 which is statistically significant at the 1% significance level. This confirms a well-known observation by Vasicek (1973) that historical beta underestimates future beta for low beta stocks and overestimates future beta for high beta stocks. We find that Bloomberg beta results in a significantly negative realized beta with a t-statistic of -1.8 which is statistically significant at the 10% level, while XGBoost constructed market neutral portfolio shows realized beta not statistically different from zero (t-stat -0.2).

A pair of robustness tests confirms the above results. When constructing long–short portfolios based on quintiles instead of deciles of historical beta, we obtain nearly identical results. We also constructed long–short portfolios based on XGBoost beta forecast sorting instead of historical beta sorting. Market neutral portfolio construction based on both historical and Bloomberg beta leads to the realized beta of the long–short portfolios being significantly different from zero at a 1% significance level. Using the XGBoost-based adjustment leads an average realized beta of the long–short portfolio that is statistically insignificantly different from zero.



Description: Stocks are sorted into deciles based on historical beta. Long sub-portfolio equal weight stocks in the first deciles (lowest beta). Short sub-portfolio equal weight stocks in the tenth deciles (highest beta). Weights of the long portion add up to 1, and weights of the short portion add up to the ratio of the beta of the first decile to that of the tenth decile. This ratio is computed based on the historical beta (blue), Bloomberg adjusted beta (orange) and XGBoost beta (grey). We plot the realized 12-month betas for each of the three long-short portfolios.

Figure 7.17. *Realized beta of long-short betting-against-beta portfolios. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

7.6. Concluding remarks

In this chapter, we applied machine learning methods to fundamental equity variables and Big Data equity sentiment variables to forecast equity beta. We found that machine learning algorithms are better at forecasting future stock beta than linear models. Big Data variables such as stock level sentiment and news volume are significant in several models in addition to other fundamental variables.

There are several avenues for future research based on our results. First, we used a relatively limited number of features to forecast beta compared to the long list of predictors used to forecast expected returns. A larger set of predictor variables could be expected to improve beta forecasts. Second, our list of algorithms did not include neural networks and other deep learning

algorithms. These algorithms have been shown to perform well in other contexts and might also improve the forecasting ability in our application.

7.7. References

Ackermann C., McEnally R. and Ravenscraft D. (1999). The performance of hedge funds: Risk, return and incentives. *Journal of Finance*, 54(3): 833–874.

Agarwal V., Daniel N.D. and Naik N.Y. (2009). Role of managerial incentives and discretion in hedge fund performance. *Journal of Finance*, 64(5): 2221–2256.

Cenesizoglu, T., Liu, Q., Reeves J.J., Wu H. (2014). Monthly beta forecasting with low, medium and high frequency stock returns. Research Paper No. 2013 BFIN 07 and FIRN Research Paper, UNSW Australian Schoolof Business. Available: https://ssrn.com/abstract=2321522.

Chang B.Y., Christoffersen P., Jacobs K. and Vainberg G. (2009). Option-impliedmeasures of equity risk. *Review of Finance*, 16(2). Available at: https://ssrn.com/abstract=1416753.

Chew D.T., Li H., Ma C., Montago C., Procacci P.F. (2019). Searching for alpha: Machine learning. *Citi Research*, Equities, New York.

Cosemans M., Frehen R., Schotman P.C. and Bauer R. (2016). Estimating security betas using prior information based on firm fundamentals. *Review of Financial Studies*, 29, 1072–1112.

Frazzini A. and Pedersen, L.H. (2014). Betting against beta. *Journal of Financial Economics*, 111, 1–25

Gu S., Kelly B.T. and Xiu D. (2019). Empirical asset pricing via machine learning. Chicago Booth Research Paper No. 18-04, 31st Australasian Finance and Banking Conference 2018. Available at: https://ssrn.com/abstract=3159577.

Hafez P. and Xie J. (2016). News beta: Factoring sentiment risk into quant models. *The Journal of Investing*, 25 (3) 88–104.

Hollstein F., Prokopczuk M., Simen C.W. (2018). Estimating beta: Forecast adjustmentsand the impact of stock characteristics for a broad cross-section. *Journalof Financial Markets*, 44, 91–118. Available at: https://ssrn.com/abstract=3069518.

Hooper V.J., Ng K. and Reeves J.J. (2008). Quarterly beta forecasting: An evaluation. *International Journal of Forecasting*, 24(3). Available at: https://ssrn.com/abstract=1367443.

Lundberg S.M. and Lee S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 4765–4774.

Lundberg, S.M.,Erion G.G. and Lee S.-I. (2018). Consistent individualized feature attribution for tree ensembles.*arXiv preprint*, arXiv:1802.03888.

Shapley L.S. (1953). A value for n-person games. In *Contributions to the Theory of Games AM-28*, Kuhn H.W. and Tucker A.W. (eds). Princeton University Press, New Jersey.

Vasicek O.A. (1973). A note on using cross-sectional information in Bayesian estimation on security betas. *The Journal of Finance*, 28(5), 1233–1239.

8

Machine Learning Optimization Algorithms & Portfolio Allocation

Portfolio optimization emerged with the seminal paper of Markowitz (1952). The original mean-variance framework is appealing because it is very efficient from a computational point of view. However, it also has one well-established failing since it can lead to portfolios that are not optimal from a financial point of view (Michaud 1989). Nevertheless, very few models have succeeded in providing a real alternative solution to the Markowitz model. The main reason lies in the fact that most academic portfolio optimization models are intractable in real life although they present solid theoretical properties. By intractable we mean that they can be implemented for an investment universe with a small number of assets, using a lot of computational resources and skills, but they are unable to manage a universe with dozens or hundreds of assets. However, the emergence and the rapid development of robo-advisors means that we need to rethink portfolio optimization and go beyond the traditional mean-variance optimization approach.

Another industry and branch of science has faced similar issues concerning large-scale optimization problems. Machine learning and applied statistics have long been associated with linear and logistic regression models. Again, the reason was the inability of optimization algorithms to solve high-dimensional industrial problems. Nevertheless, the end of the 1990s marked an important turning point with the development and the rediscovery of several methods that have since produced impressive results. The goal of this chapter is to show how portfolio allocation can benefit from the development of these large-scale optimization algorithms. Not all

---

Chapter written by Sarah PERRIN and Thierry RONCALLI.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

of these algorithms are useful in our case, but four of them are essential when solving complex portfolio optimization problems. These four algorithms are the coordinate descent, the alternating direction method of multipliers, the proximal gradient method and Dykstra's algorithm. This chapter reviews them and shows how they can be implemented in portfolio allocation.

8.1. Introduction

The contribution of Markowitz to economics is considerable. The mean-variance optimization framework marks the beginning of portfolio allocation in finance. In addition to the seminal paper of 1952, Markowitz proposed an algorithm for solving quadratic programming problems in 1956. At that time, very few people were aware of this optimization framework. We can cite Mann (1943), but it is widely accepted that Markowitz is the "father of quadratic programming" (Cottle and Infanger 2010). This is not the first time that economists are participating in the development of mathematics[1], but this is certainly the first time that mathematicians will explore a field of research whose main application during the first years of research is exclusively an economic problem.

The success of mean-variance optimization (MVO) is due to the appealing properties of the quadratic utility function, but it should also be assessed in light of the success of quadratic programming (QP). Because it is easy to solve QP problems and because QP problems are available in mathematical software, solving MVO problems is straightforward and does not require a specific skill. Hence, the mean-variance optimization is a universal method which is used by all portfolio managers. However, this approach has been widely criticized by academics and professionals. Indeed, mean-variance optimization is very sensitive to input parameters and produces corner solutions. Moreover, the concept of mean-variance diversification is confused with the concept of hedging (Bourgeron *et al.* 2018). These different issues make the practice of mean-variance optimization less attractive than the theory (Michaud 1989). In fact, solving MVO allocation problems requires

1 For example, Leonid Kantorovich made major contributions to the success of linear programming.

the right weight constraints to be specified, in order to obtain acceptable solutions. It follows that designing the constraints is the most important component of mean-variance optimization. In this case, MVO appears to be a trial-and-error process, not a systematic solution.

The success of the MVO framework is also explained by the fact that there are very few competing portfolio allocation models that can be implemented from an industrial point of view. There are generally two reasons for this. The first one is that some models use input parameters that are difficult to estimate or understand, making these models definitively unusable. The second reason is that other models use a more complex objective function than the simple quadratic utility function. In this case, the computational complexity makes these models less attractive than the standard MVO model. Among these models, some of them are based on the mean-variance objective function, but introduce regularization penalty functions, in order to improve the robustness of the portfolio allocation. Again, these models have little chance of being used if they cannot be cast into a QP problem. However, new optimization algorithms have emerged for solving large-scale machine learning problems. The purpose of this chapter is to present these new mathematical methods and show that they can be easily applied to portfolio allocation in order to go beyond the MVO/QP model.

This chapter is based on several previous research papers (Bourgeron *et al*. 2018, Griveau-Billion *et al*. 2013, Richard and Roncalli 2015 and Richard and Roncalli 2019) and extensively uses four leading references (Beck 2017; Boyd *et al.* 2010; Combettes and Pesquet 2011; Tibshirani 2017). It is organized as follows. In section 8.2, we present the mean-variance approach and how it is related to the QP framework. The third section is dedicated to large-scale optimization algorithms that have been used in machine learning: coordinate descent, alternating direction method of multipliers, proximal gradient and Dykstra's algorithm. Section 8.4 shows how these algorithms can be implemented in order to solve portfolio optimization problems and build a more robust asset allocation. Finally, section 8.5 offers some concluding remarks.

8.2. The quadratic programming world of portfolio optimization

8.2.1. *Quadratic programming*

8.2.1.1. *Primal and dual formulation*

A quadratic programming (QP) problem is an optimization problem with a quadratic objective function and linear inequality constraints:

$$x^\star = \arg\min_x \frac{1}{2} x^\top Q x - x^\top R \qquad \text{s.t.} \quad Sx \leq T \tag{8.1}$$

where x is an $n \times 1$ vector, Q is an $n \times n$ matrix and R is an $n \times 1$ vector. We note that the system of constraints $Sx \leq T$ allows us to specify linear equality constraints[2] $Ax = B$ or box constraints $x^- \leq x \leq x^+$. Most numerical packages then consider the following formulation:

$$x^\star = \arg\min_x \frac{1}{2} x^\top Q x - x^\top R \tag{8.2}$$

$$\text{s.t.} \begin{cases} Ax = B \\ Cx \leq D \\ x^- \leq x \leq x^+ \end{cases}$$

because the problem [8.2] is equivalent to the canonical problem [8.1] with the following system of linear inequalities: $-Ax \leq -B$, $Ax \leq B$, $Cx \leq D$, $-I_n x \leq -x^-$ and $I_n x \leq x^+$. If the space Ω defined by $Sx \leq T$ is non-empty and if Q is a symmetric positive definite matrix, the solution exists because the function $f(x) = \frac{1}{2} x^\top Q x - x^\top R$ is convex. In the general case where Q is a square matrix, the solution may not exist.

The QP problem has the interesting property that the dual formulation is another quadratic programming problem:

$$\lambda^\star = \arg\min_\lambda \frac{1}{2} \lambda^\top \bar{Q} \lambda - \lambda^\top \bar{R} \qquad \text{s.t.} \quad \lambda \geq 0 \tag{8.3}$$

where $\bar{Q} = SQ^{-1}S^\top$ and $\bar{R} = SQ^{-1}R - T$. This duality property is very important for some machine learning methods. For example, this is the case of

2 This is equivalent to impose that $Ax \geq B$ and $Ax \leq B$.

support vector machines and kernel methods that extensively use the duality for defining the solution (Cortes and Vapnik 1995).

8.2.1.2. *Numerical algorithms*

There is a substantial literature on the methods for solving quadratic programming problems (Gould and Toint 2000). The research begins in the 1950s with different key contributions: Frank and Wolfe (1956), Markowitz (1956), Beale (1959) and Wolfe (1959). Nowadays, QP problems are generally solved using three approaches: active set methods, gradient projection methods and interior point methods. All these algorithms are implemented in standard mathematical programming languages (MATLAB, Matematica, Python, Gauss, R, etc.). This explains the success of QP problems since the 2000s, because they can be easily and rapidly solved.

8.2.2. *Mean-variance optimized portfolios*

The concept of portfolio allocation has a long history and dates back to the seminal work of Markowitz (1952). In his paper, Markowitz defined precisely what *portfolio selection* means: "the investor does (or should) consider expected return a desirable thing and variance of return an undesirable thing". Indeed, Markowitz showed that an efficient portfolio is the portfolio that maximizes the expected return for a given level of risk (corresponding to the variance of portfolio return) or a portfolio that minimizes the risk for a given level of expected return. Even if this framework has been extended to many other allocation problems (index sampling, turnover management, etc.), the mean-variance model remains the optimization approach that is most widely used in finance.

8.2.2.1. *The Markowitz framework*

We consider a universe of n assets. Let $x = (x_1, \ldots, x_n)$ be the vector of weights in the portfolio. We assume that the portfolio is fully invested meaning that $\sum_{i=1}^{n} x_i = \mathbf{1}_n^\top x = 1$. We denote $\mathfrak{R} = (\mathfrak{R}_1, \ldots, \mathfrak{R}_n)$ as the vector of asset returns where $\mathfrak{R}_i$ is the return of asset i. The return of the portfolio is then equal to $\mathfrak{R}(x) = \sum_{i=1}^{n} x_i \mathfrak{R}_i = x^\top \mathfrak{R}$. Let $\mu = \mathbb{E}[\mathfrak{R}]$ and $\Sigma = \mathbb{E}\left[(\mathfrak{R} - \mu)(\mathfrak{R} - \mu)^\top\right]$ be the vector of expected returns and the covariance matrix of asset returns, respectively. The expected return of the portfolio is equal to $\mu(x) = \mathbb{E}[\mathfrak{R}(x)] = x^\top \mu$, whereas its variance is equal to

$\sigma^2(x) = \mathbb{E}\left[(\Re(x) - \mu(x))(\Re(x) - \mu(x))^\top\right] = x^\top \Sigma x$. Markowitz (1952) formulated the investor's financial problem as follows:

1) maximizing the expected return of the portfolio under a volatility constraint:

$$\max \mu(x) \quad \text{s.t.} \quad \sigma(x) \leq \sigma^\star \tag{8.4}$$

2) or minimizing the volatility of the portfolio under a return constraint:

$$\min \sigma(x) \quad \text{s.t.} \quad \mu(x) \geq \mu^\star \tag{8.5}$$

Markowitz's bright idea was to consider a quadratic utility function $\mathcal{U}(x) = x^\top \mu - \frac{\phi}{2} x^\top \Sigma x$, where $\phi \geq 0$ is the risk aversion. Since maximizing $\mathcal{U}(x)$ is equivalent to minimizing $-\mathcal{U}(x)$, the Markowitz problems [8.4] and [8.5] can be cast into a QP problem[3]:

$$x^\star(\gamma) = \arg\min_x \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu \qquad \text{s.t.} \quad \mathbf{1}_n^\top x = 1 \tag{8.6}$$

where $\gamma = \phi^{-1}$. Therefore, solving the μ-problem or the σ-problem is equivalent to finding the optimal value of γ such that $\mu(x^\star(\gamma)) = \mu^\star$ or $\sigma(x^\star(\gamma)) = \sigma^\star$. We know that the functions $\mu(x^\star(\gamma))$ and $\sigma(x^\star(\gamma))$ are increasing with respect to γ and are bounded. The optimal value of γ can then be easily computed using the bisection algorithm. It is obvious that a large part of the success of the Markowitz framework lies on the QP trick. Indeed, problem [8.6] corresponds to the QP problem [8.2], where $Q = \Sigma$, $R = \gamma\mu$, $A = \mathbf{1}_n^\top$ and $B = 1$. Moreover, it is easy to include bounds on the weights, inequalities between asset classes, etc.

8.2.2.2. *Solving complex MVO problems*

The previous framework can be extended to other portfolio allocation problems. However, from a numerical point of view, the underlying idea is to always find an equivalent QP formulation (Roncalli 2013).

8.2.2.2.1. Portfolio optimization with a benchmark

We now consider a benchmark b. We note $\mu(x \mid b) = (x - b)^\top \mu$ as the expected excess return and $\sigma(x \mid b) = \sqrt{(x - b)^\top \Sigma (x - b)}$ as the tracking

3 This transformation is called the QP trick.

error volatility of Portfolio x with respect to Benchmark b. The objective function corresponds to a trade-off between minimizing the tracking error volatility and maximizing the expected excess return (or the alpha):

$$f\left(x \mid b\right) = \frac{1}{2}\sigma^2\left(x \mid b\right) - \gamma\mu\left(x \mid b\right)$$

We can show that the equivalent QP problem is:

$$x^\star\left(\gamma\right) = \arg\min_x \frac{1}{2}x^\top \Sigma x - \gamma x^\top \tilde{\mu}$$

where $\tilde{\mu} = \mu + \gamma^{-1}\Sigma b$ is the regularized vector of expected returns. Therefore, portfolio allocation with a benchmark can be viewed as a regularization of the MVO problem and is solved using a QP numerical algorithm.

8.2.2.2.2. Index sampling

The goal of index sampling is to replicate an index portfolio with a smaller number of assets than the index (or the benchmark) b. From a mathematical point of view, index sampling could be written as follows:

$$x^\star = \arg\min_x \frac{1}{2}\left(x - b\right)^\top \Sigma \left(x - b\right) \quad [8.7]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ x \geq \mathbf{0}_n \\ \sum_{i=1}^n \quad \left\{x_i > 0\right\} \leq n_x \end{cases}$$

The idea is to minimize the volatility of the tracking error such that the number of stocks n_x in the portfolio is smaller than the number of stocks n_b in the benchmark. For example, we would like to replicate the S&P 500 index with only 50 stocks and not the entire 500 stocks that compose this index. Professionals generally solve problem [8.7] with the following heuristic algorithm:

1) We set $x_{(0)}^{+} = \mathbf{1}_n$. At the iteration $k+1$, we solve the QP problem:

$$x^\star = \arg\min_x \frac{1}{2}(x-b)^\top \Sigma (x-b)$$
$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x_{(k)}^{+} \end{cases}$$

2) We then update the upper bound $x_{(k)}^{+}$ of the QP problem by deleting the asset $i^\star$ with the lowest non-zero optimized weight[4]:

$$x_{(k+1),i}^{+} \leftarrow x_{(k),i}^{+} \quad \text{if } i \neq i^\star \quad \text{and} \quad x_{(k+1),i^\star}^{+} \leftarrow 0$$

3) We iterate the two steps until $\sum_{i=1}^{n} \quad \{x_i^\star > 0\} = n_x$.

The purpose of the heuristic algorithm is to delete one asset at each iteration in order to obtain an invested portfolio, which is composed exactly of n_x assets and has a low tracking error volatility. Again, we note that solving the index sampling problem is equivalent to solving $(n_b - n_x)$ QP problems.

8.2.2.2.3. Turnover management

If we note $\bar{x}$ as the current portfolio and x as the new portfolio, the turnover of portfolio x with respect to portfolio $\bar{x}$ is the sum of purchases and sales:

$$\boldsymbol{\tau}(x \mid \bar{x}) = \sum_{i=1}^{n} (x_i - \bar{x}_i)^+ + \sum_{i=1}^{n} (\bar{x}_i - x_i)^+ = \sum_{i=1}^{n} |x_i - \bar{x}_i|$$

Adding a turnover constraint in long-only MVO portfolios leads to the following problem:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu$$
$$\text{s.t.} \begin{cases} \sum_{i=1}^{n} x_i = 1 \\ \sum_{i=1}^{n} |x_i - \bar{x}_i| \leq \boldsymbol{\tau}^+ \\ 0 \leq x_i \leq 1 \end{cases}$$

4 We have $i^\star = \{ i : \arg\inf x_i^\star \,|\, x_i^\star > 0 \}$.

where $\boldsymbol{\tau}^+$ is the maximum turnover with respect to the current portfolio $\bar{x}$. Scherer (2007) introduces the additional variables x_i^- and x_i^+ such that $x_i = \bar{x}_i + x_i^+ - x_i^-$, where $x_i^- \geq 0$ indicates a negative weight change with respect to the initial weight $\bar{x}_i$ and $x_i^+ \geq 0$ indicates a positive weight change. The expression of the turnover becomes:

$$\sum_{i=1}^{n} |x_i - \bar{x}_i| = \sum_{i=1}^{n} \left| x_i^+ - x_i^- \right| = \sum_{i=1}^{n} x_i^+ + \sum_{i=1}^{n} x_i^-$$

because one of the variables x_i^+ or x_i^- is necessarily equal to zero due to the minimization problem. The γ-problem of Markowitz becomes:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2} x^\top \Sigma x - \gamma x^\top \mu \\ \text{s.t.} &\begin{cases} \sum_{i=1}^n x_i = 1 \\ x_i = \bar{x}_i + x_i^+ - x_i^- \\ \sum_{i=1}^n x_i^+ + \sum_{i=1}^n x_i^- \leq \boldsymbol{\tau}^+ \\ 0 \leq x_i, x_i^-, x_i^+ \leq 1 \end{cases} \end{aligned}$$

We obtain an augmented QP problem of dimension $3n$ (see Appendix 8.7.1.1).

8.2.2.2.4. Transaction costs

The previous analysis assumes that there is no transaction cost $\boldsymbol{c}(x \mid \bar{x})$ when we rebalance the portfolio from the current portfolio $\bar{x}$ to the new optimized portfolio x. If we note c_i^- and c_i^+ as the bid and ask transaction costs, then we have:

$$\boldsymbol{c}(x \mid \bar{x}) = \sum_{i=1}^{n} x_i^- c_i^- + \sum_{i=1}^{n} x_i^+ c_i^+$$

The net expected return of portfolio x is then equal to $\mu(x) - c(x \mid \bar{x})$. It follows that the γ-problem of Markowitz becomes[5]:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x - \gamma \left(\sum_{i=1}^n x_i \mu_i - \sum_{i=1}^n x_i^- c_i^- - \sum_{i=1}^n x_i^+ c_i^+ \right)$$

$$\text{s.t.} \begin{cases} \sum_{i=1}^n x_i + \sum_{i=1}^n x_i^- c_i^- + \sum_{i=1}^n x_i^+ c_i^+ = 1 \\ x_i = \bar{x}_i + x_i^+ - x_i^- \\ 0 \leq x_i, x_i^-, x_i^+ \leq 1 \end{cases}$$

Once again, we obtain a QP problem of dimension $3n$.

8.2.3. *Issues with QP optimization*

The concurrent model of the Markowitz framework is the risk budgeting approach (Qian 2005; Maillard *et al.* 2010; Roncalli 2013). The goal is to define a convex risk measure $\mathcal{R}(x)$ and to allocate the risk according to some specified risk budgets $\mathcal{RB} = (\mathcal{RB}_1, \ldots, \mathcal{RB}_n)$, where $\mathcal{RB}_i > 0$. This approach exploits the Euler decomposition property of the risk measure:

$$\mathcal{R}(x) = \sum_{i=1}^n x_i \frac{\partial \mathcal{R}(x)}{\partial x_i}$$

By noting $\mathcal{RC}_i(x) = x_i \cdot \partial_{x_i} \mathcal{R}(x)$ as the risk contribution of asset i with respect to portfolio x, the risk budgeting (RB) portfolio is defined by the following set of equations: $x^\star = \{x \in [0,1]^n : \mathcal{RC}_i(x) = \mathcal{RB}_i\}$. Roncalli (2013) showed that it is equivalent to solving the following nonlinear optimization problem[6]:

$$x^\star = \arg\min_x \mathcal{R}(x) - \lambda \sum_{i=1}^n \mathcal{RB}_i \cdot \ln x_i \qquad [8.8]$$

5 The equality constraint $1_n^\top x = 1$ becomes $1_n^\top x + c(x \mid \bar{x}) = 1$ because the rebalancing process has to be financed.

6 In fact, the solution $x^\star$ must be rescaled after the optimization step.

where $\lambda > 0$ is an arbitrary positive constant. Generally, the most frequently used risk measures are the volatility risk measure $\mathcal{R}(x) = \sqrt{x^\top \Sigma x}$ (Maillard *et al.* 2010) and the standard deviation-based risk measure $\mathcal{R}(x) = -x^\top (\mu - r) + \xi\sqrt{x^\top \Sigma x}$, where r is the risk-free rate and ξ is a positive scalar (Roncalli 2015). In particular, this last one encompasses the Gaussian value-at-risk – $\xi = \Phi^{-1}(\alpha)$ – and the Gaussian expected shortfall – $\xi = (1-\alpha)^{-1}\phi\left(\Phi^{-1}(\alpha)\right)$.

The risk budgeting approach has displaced the MVO approach in many fields of asset management, in particular in the case of factor investing and alternative risk premia. Nevertheless, we note that problem [8.8] is a not a quadratic programming problem, but a logarithmic barrier problem. Therefore, the risk budgeting framework opens a new world of portfolio optimization that is not necessarily QP! That is all the more true since MVO portfolios face robustness issues (Bourgeron *et al.* 2018). Regularization of portfolio allocation has then become the industry standard. Indeed, it is frequent to add an ℓ_1-norm or ℓ_2-norm penalty function to the MVO objective function. This type of penalty is, however, tractable in a quadratic programming setting. With the development of robo-advisors, nonlinear penalty functions have emerged, in particular the logarithmic barrier penalty function. And these regularization techniques result in a non-quadratic programming world of portfolio optimization.

The success of this non-QP financial world will depend on how quickly and easily these complex optimization problems can be solved. Griveau-Billon *et al.* (2013), Bourgeron *et al.* (2018) and Richard and Roncalli (2019) have already proposed numerical algorithms that are doing the work in some special cases. The next section reviews the candidate algorithms that may compete with QP numerical algorithms.

8.3. Machine learning optimization algorithms

The machine learning industry has experienced a similar trajectory to portfolio optimization. Before the 1990s, statistical learning focused mainly on models that were easy to solve from a numerical point of view. For instance, the linear (and the ridge) regression has an analytical solution, we can solve logistic regression with the Newton–Raphson algorithm, whereas

supervised and unsupervised classification models[7] consist of performing a singular value decomposition or a generalized eigenvalue decomposition. The 1990s saw the emergence of three models that have deeply changed the machine learning approach: neural networks, support vector machines and lasso regression.

Neural networks have been extensively studied since the seminal work of Rosenblatt (1958). However, the first industrial application dates back to the publication of LeCun *et al.* (1989) on handwritten zip code recognition. At the beginning of the 1990s, a fresh craze then emerged with the writing of many handbooks that were appropriate for students, the most popular of which was Bishop (1995). With neural networks, two main issues arise concerning calibration: the large number of parameters to estimate and the absence of a global maximum. The traditional numerical optimization algorithms[8] that were popular in the 1980s cannot be applied to neural networks. New optimization approaches are then proposed. First, researchers have considered more complex learning rules than the steepest descent (Jacobs 1988), for example, the momentum method of Polyak (1964) or the Nesterov accelerated gradient approach (Nesterov 1983). Second, the descent method is generally not performed on the full sample of observations, but on a subset of observations that changes at each iteration. This is the underlying idea behind batch gradient descent (BGD), stochastic descent gradient (SGD) and mini-batch gradient descent (MGD). We note that adaptive learning methods and batch optimization techniques have marked the revival of the gradient descent method.

The development of support vector machines is another important step in the development of machine learning techniques. Like neural networks, they can be seen as an extension of the perceptron. However, they present nice theoretical properties and a strong geometrical framework. Once SVMs have been first developed for linear classification, they have been extended for nonlinear classification and regression. A support vector machine consists of separating hyperplanes and finding the optimal separation by maximizing the

7 For example, principal component analysis (PCA), linear/quadratic discriminant analysis (LDA/QDA) and Fisher classification method.

8 For example, we can cite the quasi-Newton BFGS (Broyden–Fletcher–Goldfarb–Shanno) and DFP (Davidon–Fletcher–Powell) methods and the Fletcher–Reeves and Polak–Ribiere conjugate gradient methods.

margin. The original problem called the hard margin classification can be formulated as a quadratic programming problem. However, the dual problem, which is also a QP problem, is generally preferred to the primal problem for solving SVM classification, because of the sparse property of the solution (Cortes and Vapnik 1995). Over the years, the original hard margin classification has been extended to other problems: soft margin classification with binary hinge loss, soft margin classification with squared hinge loss, least squares SVM regression, ε-SVM regression, kernel machines (Vapnik 1998). All these statistical problems share the same calibration framework. The primal problem can be cast into a QP problem, implying that the corresponding dual problem is also a QP problem. Again, we note that the success and the prominence of statistical methods are related to the efficiency of the optimization algorithms and it is obvious that support vector machines have substantially benefited from the QP formulation. From an industrial point of view however, support vector machines present some limitations. Indeed, if the dimension of the primal QP problem is the number p of features (or parameters), the dimension of the dual QP problem is the number n of observations. It is becoming absolutely impossible to solve the dual problem when the number of observations is larger than 100,000 and sometimes as high as several millions. This implies that new algorithms that are more appropriate for large-scale optimization problems need to be developed.

Lasso regression is the third disruptive approach that put machine learning in the spotlight in the 1990s. Like the ridge regression, lasso regression is a regularized linear regression where the $\boldsymbol{\ell}_2$-norm penalty is replaced by the $\boldsymbol{\ell}_1$-norm penalty (Tibshirani 1996). Since the $\boldsymbol{\ell}_1$ regularization forces the solution to be sparse, it has been largely used first for variable selection and then for pattern recognition and robust estimation of linear models. For finding the lasso solution, the technique of augmented QP problems is widely used since it is easy to implement. The extension of the lasso-ridge regularization to the other $\boldsymbol{\ell}_p$ norms is straightforward but these approaches have never been popular. The main reason is that existing numerical algorithms are not sufficient to make these models tractable.

Therefore, the success of a quantitative model may be explained by two conditions. First, the model must be obviously appealing. Second, the model must be solved by an efficient numerical algorithm that is easy to implement or available in mathematical programming software. As shown previously, quadratic programming and gradient descent methods have been key for

many statistical and financial models. In what follows, we consider four algorithms and techniques that have been popularized by their use in machine learning: coordinate descent, alternating direction method of multipliers, proximal operators and Dykstra's algorithm. In particular, we illustrate how they can be used for solving complex optimization problems.

8.3.1. *Coordinate descent*

8.3.1.1. *Definition*

We consider the following unconstrained minimization problem:

$$x^{\star} = \arg\min_{x} f(x) \qquad [8.9]$$

where $x \in \mathbb{R}^n$ and $f(x)$ is a continuous, smooth and convex function. A popular method to find the solution $x^{\star}$ is to consider the descent algorithm, which is defined by the following rule:

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)} = x^{(k)} - \eta D^{(k)}$$

where $x^{(k)}$ is the approximated solution of problem [8.9] at the k^{th} iteration, $\eta > 0$ is a scalar that determines the step size and $D^{(k)}$ is the direction. We note that the current solution $x^{(k)}$ is updated by going in the opposite direction to $D^{(k)}$ in order to obtain $x^{(k+1)}$. In the case of the gradient descent, the direction is equal to the gradient vector of $f(x)$ at the current point: $D^{(k)} = \nabla f\left(x^{(k)}\right)$. Coordinate descent (CD) is a variant of the gradient descent and minimizes the function along one coordinate at each step:

$$x_i^{(k+1)} = x_i^{(k)} + \Delta x_i^{(k)} = x_i^{(k)} - \eta D_i^{(k)}$$

where $D_i^{(k)} = \nabla_i f\left(x^{(k)}\right)$ is the i^{th} element of the gradient vector. At each iteration, a coordinate i is then chosen via a certain rule, while the other coordinates are assumed to be fixed. Coordinate descent is an appealing algorithm because it transforms a vector-valued problem into a scalar-valued problem that is easier to implement. Another formulation of the coordinate descent method is to replace the descent approximation by the exact problem.

Indeed, the objective of the descend step is to minimize the scalar-valued problem:

$$x_i^\star = \arg\min_{\varkappa} f\left(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, \varkappa, x_{i+1}^{(k)}, \ldots, x_n^{(k)}\right) \quad [8.10]$$

Coordinate descent is efficient in large-scale optimization problems, in particular when there is a solution to the scalar-valued problem [8.10]. Furthermore, convergence is guaranteed when $f(x)$ is convex and differentiable.

REMARK 8.1.– Coordinate descent methods were introduced in several handbooks on numerical optimization in the 1980s and 1990s (Wright 2015). However, the most important step is the contribution of Tseng (2001), who studied the block-coordinate descent method and extended CD algorithms in the case of a non-differentiable and non-convex function $f(x)$.

8.3.1.2. *Cyclic or random coordinates?*

There are several options for choosing the coordinate of the k^{th} iteration. A natural choice could be to choose the coordinate, which minimizes the function:

$$i^\star = \arg\inf\left\{f_i^\star : i \in \{1, n\}, f_i^\star = \min_{\varkappa} f\left(\left(1 - e_i\right) x^{(k)} + e_i \varkappa\right)\right\}$$

However, it is obvious that choosing the optimal coordinate $i^\star$ would require the gradient along each coordinate to be calculated. This causes the coordinate descent to be no longer efficient, since a classic gradient descent would then be of equivalent cost at each iteration and would converge faster because it requires fewer iterations.

The simplest way to implement the CD algorithm is to consider cyclic coordinates, meaning that we cyclically iterate through the coordinates (Tseng 2001): $i = k \bmod n$. This ensures that all the coordinates are selected during one cycle $\{k - n + 1, \ldots, k\}$ in the same order. This approach, called cyclical coordinate descent (CCD), is the most popular and most used method, even if it is difficult to estimate the rate of convergence.

The second way is to consider random coordinates. Let π_i be the probability of choosing the coordinate i at the iteration k. The simplest approach is to consider uniform probabilities: $\pi_i = 1/n$. A better approach

consists of pre-specifying probabilities according to the Lipschitz constants[9]: $\pi_i = \mathfrak{L}_i^\alpha / \sum_{j=1}^n \mathfrak{L}_j^\alpha$. Nesterov (2012) considers three schemes: $\alpha = 0$, $\alpha = 1$ and $\alpha = \infty$ – in this last case, we have $i = \arg\max \{\mathfrak{L}_1, \ldots, \mathfrak{L}_n\}$. From a theoretical point of view, the random coordinate descent (RCD) method based on the probability distribution π leads to a faster convergence, since coordinates that have a large Lipschitz constant $\mathfrak{L}_i$ are more likely to be chosen. However, it requires additional calculus to compute the Lipschitz constants and CCD is often preferred from a practical point of view. In what follows, we only use the CCD algorithm described below. In Algorithm [8.1.], the variable k represents the number of cycles, whereas the number of iterations is equal to $k \cdot n$. For the coordinate i, the lower coordinates $j < i$ correspond to the current cycle $k+1$, while the upper coordinates $j > i$ correspond to the previous cycle k.

Algorithm 8.1. Cyclical coordinate descent algorithm

The goal is to find the solution $x^\star = \arg\min f(x)$
We initialize the vector $x^{(0)}$
Set $k \leftarrow 0$
repeat
  for $i = 1 : n$ **do**
    $x_i^{(k+1)} = \arg\min_\varkappa f\left(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, \varkappa, x_{i+1}^{(k)}, \ldots, x_n^{(k)}\right)$
  end for
  $k \leftarrow k+1$
until convergence
return $x^\star \leftarrow x^{(k)}$

8.3.1.3. *Application to the λ-problem of the lasso regression*

We consider the linear regression $Y = X\beta + \varepsilon$, where Y is the $n \times 1$ vector, X is the $n \times p$ design matrix, β is the $p \times 1$ vector of coefficients and ε is the $n \times 1$ vector of residuals. In this model, n is the number of observations and p is the number of parameters (or the number of explanatory variables). The objective of the ordinary least squares is to minimize the residual sum of squares $\hat{\beta} = \arg\min_\beta \frac{1}{2} \mathrm{RSS}(\beta)$, where $\mathrm{RSS}(\beta) =$

9 Nesterov (2012) assumes that $f(x)$ is convex, differentiable and Lipschitz smooth for each coordinate: $\|\nabla_i f(x + e_i h) - \nabla_i f(x)\| \leq \mathfrak{L}_i \|h\|$, where $h \in \mathbb{R}$.

$\sum_{i=1}^{n} \varepsilon_i^2$. Since we have $\text{RSS}(\beta) = (Y - X\beta)^\top (Y - X\beta)$, we obtain $\partial_{\beta_j} f(\beta) = -x_j^\top (Y - X\beta)$, where x_j is the $n \times 1$ design vector corresponding to the j^{th} explanatory variable. Because we can write $X\beta = X_{(-j)}\beta_{(-j)} + x_j\beta_j$, where $X_{(-j)}$ and $\beta_{(-j)}$ are the design matrix and the beta vector by excluding the j^{th} explanatory variable, respectively, it follows that $\partial_{\beta_j} f(\beta) = x_j^\top X_{(-j)}\beta_{(-j)} + x_j^\top x_j\beta_j - x_j^\top Y$. At the optimum, we have $\partial_{\beta_j} f(\beta) = 0$ or:

$$\beta_j = \frac{x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right)}{x_j^\top x_j} \quad [8.11]$$

The implementation of the coordinate descent algorithm is straightforward. It suffices to iterate equation [8.11] through the coordinates.

The lasso regression problem is a variant of the OLS regression by adding an $\boldsymbol{\ell}_1$-norm regularization (Tibshirani 1996):

$$\hat{\beta}(\lambda) = \arg\min_{\beta} \frac{1}{2} (Y - X\beta)^\top (Y - X\beta) + \lambda \|\beta\|_1 \quad [8.12]$$

In this formulation, the residual sum of squares of the linear regression is penalized by a term that will force a sparse selection of the coordinates. Since the objective function is the sum of two convex norms, the convergence is guaranteed for the lasso problem. Because $\|\beta\|_1 = \sum_{j=1}^{n} |\beta_j|$, the first-order condition becomes $x_j^\top x_j\beta_j - x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right) + \lambda\partial |\beta_j| = 0$. In Appendix 8.7.1.3, we show that the solution is given by:

$$\beta_j = \frac{\mathcal{S}\left(x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right); \lambda\right)}{x_j^\top x_j} \quad [8.13]$$

where $\mathcal{S}(v;\lambda)$ is the soft-thresholding operator $\mathcal{S}(v;\lambda) = \text{sign}(v) \cdot (|v| - \lambda)_+$. It follows that the lasso CD algorithm is a variation of the linear regression CD algorithm by applying the soft-threshold operator to the residuals $x_j^\top \left(Y - X_{(-j)}\beta_{(-j)}\right)$ at each iteration.

Let us consider an experiment with $n = 10,0000$ and $p = 50$. The design matrix X is built using the uniform distribution, while the residuals are simulated using a Gaussian distribution and a standard deviation of 20%. The

beta coefficients are distributed uniformly between -3 and $+3$ except four coefficients that take a larger value. We then standardize the data of X and Y because the practice of the lasso regression is to consider comparable beta coefficients. By considering uniform numbers between -1 and $+1$ for initializing the coordinates, results of the CCD algorithm are given in Figure 8.1. We note that the CCD algorithm quickly converges after three complete cycles. In the case of a large-scale problem when $p \gg 1\,000$, it has been shown that CCD may be faster for the lasso regression than for the OLS regression because of the soft-thresholding operator. Indeed, we can initialize the algorithm with the null vector $\mathbf{0}_p$. If λ is large, a lot of optimal coordinates are equal to zero and a few cycles are needed to find the optimal values of non-zero coefficients.

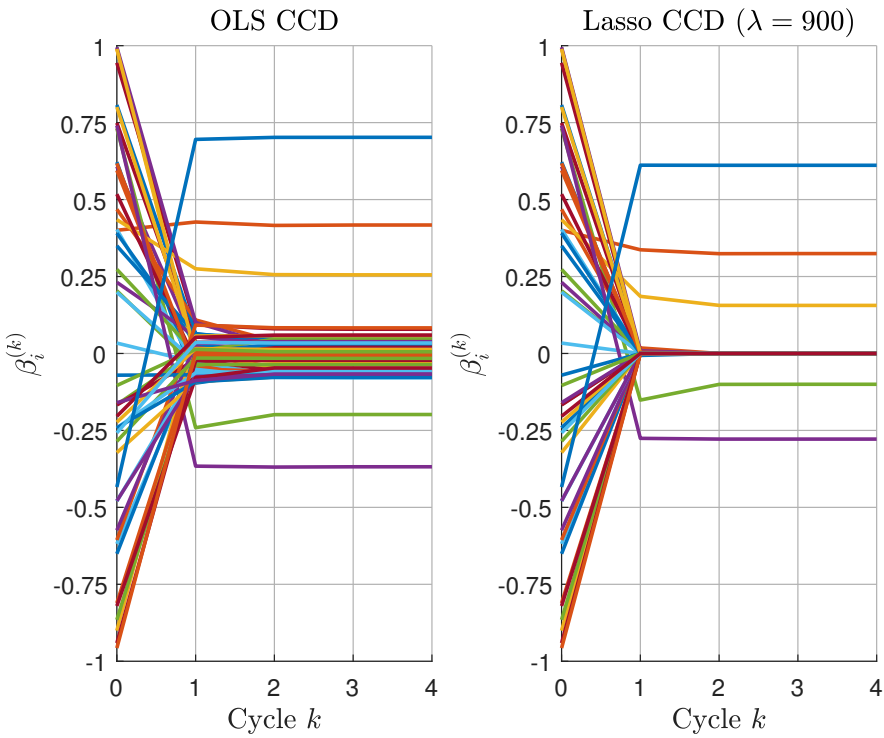


Figure 8.1. *CCD algorithm applied to the lasso optimization problem. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

8.3.2. *Alternating direction method of multipliers*

8.3.2.1. *Definition*

The alternating direction method of multipliers (ADMM) is an algorithm introduced by Gabay and Mercier (1976) to solve optimization problems, which can be expressed as:

$$\{x^\star, y^\star\} = \arg\min_{(x,y)} f_x(x) + f_y(y) \quad \text{s.t. } Ax + By = c \qquad [8.14]$$

where $A \in \mathbb{R}^{p\times n}$, $B \in \mathbb{R}^{p\times m}$, $c \in \mathbb{R}^p$ and the functions $f_x : \mathbb{R}^n \to \mathbb{R}\cup\{+\infty\}$ and $f_y : \mathbb{R}^m \to \mathbb{R}\cup\{+\infty\}$ are proper closed convex functions. Boyd *et al.* (2011) show that the ADMM algorithm consists of the following three steps:

1) the x-update is:

$$x^{(k+1)} = \arg\min_x \left\{ f_x(x) + \frac{\varphi}{2}\left\| Ax + By^{(k)} - c + u^{(k)} \right\|_2^2 \right\}; \qquad [8.15]$$

2) the y-update is:

$$y^{(k+1)} = \arg\min_y \left\{ f_y(y) + \frac{\varphi}{2}\left\| Ax^{(k+1)} + By - c + u^{(k)} \right\|_2^2 \right\}; \qquad [8.16]$$

3) the u-update is:

$$u^{(k+1)} = u^{(k)} + \left(Ax^{(k+1)} + By^{(k+1)} - c \right). \qquad [8.17]$$

In this approach, $u^{(k)}$ is the dual variable of the primal residual $r = Ax + By - c$ and φ is the $\boldsymbol{\ell}_2$-norm penalty variable. The parameter φ can be constant or may change at each iteration. The ADMM algorithm benefits from the dual ascent principle and the method of multipliers. The difference with the latter is that the x- and y-updates are performed in an alternating way. Therefore, it is more flexible because the updates are equivalent to computing proximal operators for f_x and f_y independently. In practice, ADMM may be slow to converge with high accuracy but is fast to converge if we consider modest accuracy. Hence, ADMM is a good candidate for solving large-scale machine learning problems, where high accuracy does not necessarily lead to a better solution.

REMARK 8.2.– In this chapter, we use the notations $f_x^{(k+1)}(x)$ and $f_y^{(k+1)}(y)$ when referring to the objective functions that are defined in the x- and y-updates.

8.3.2.2. *ADMM tricks*

The appeal of ADMM is that it can separate a complex problem into two sub-problems that are easier to solve. However, most of the time, the optimization problem is not formulated using a separable objective function. The question is then how to formulate the initial problem as a separable problem. We now list some tricks that show how ADMM may be used in practice.

8.3.2.2.1. First trick

We consider a problem of the form $x^\star = \arg\min_x g(x)$. The idea is then to write $g(x)$ as a separable function $g(x) = g_1(x) + g_2(x)$ and to consider the following equivalent ADMM problem:

$$\begin{aligned} \{x^\star, y^\star\} \;=\; & \arg\min_{(x,y)} f_x(x) + f_y(y) \\ & \text{s.t. } x = y \end{aligned} \qquad [8.18]$$

where $f_x(x) = g_1(x)$ and $f_y(y) = g_2(y)$. Usually, the smooth part of $g(x)$ will correspond to $g_1(x)$ while the non-smooth part will be included in $g_2(y)$. The underlying idea is that the x-update is straightforward, whereas the y-update deals with the tricky part of $g(x)$.

8.3.2.2.2. Second trick

If we want to minimize the function $g(x)$, where $x \in \Omega$ is a set of constraints, the optimization problem can be cast into the ADMM form [8.18], where $f_x(x) = g(x)$, $f_y(y) = \iota_\Omega(y)$ and $\iota_\Omega(x)$ is the convex indicator function of Ω:

$$\iota_\Omega(x) = \begin{cases} 0 & \text{if } x \in \Omega \\ +\infty & \text{if } x \notin \Omega \end{cases} \qquad [8.19]$$

For example, if we want to solve the QP problem [8.2] given on page 264, we have $f_x(x) = \frac{1}{2}x^\top Q x - x^\top R$ and $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$.

8.3.2.2.3. Third trick

We can combine the first and second tricks. For instance, if we consider the following optimization problem:

$$\begin{aligned} x^{\star} &= \arg\min_{x} g_1(x) + g_2(x) \\ &\text{s.t. } x \in \Omega_1 \cap \Omega_2 \end{aligned}$$

the equivalent ADMM form is:

$$\begin{aligned} \{x^{\star}, y^{\star}\} &= \arg\min_{(x,y)} \underbrace{(g_1(x) + \quad_{\Omega_1}(x))}_{f_x(x)} + \underbrace{(g_2(y) + \quad_{\Omega_2}(y))}_{f_y(y)} \\ &\text{s.t. } x = y \end{aligned}$$

Let us consider a variant of the QP problem where we add a nonlinear constraint $h(x) = 0$. In this case, we can write the set of constraints as $\Omega = \Omega_1 \cap \Omega_2$, where $\Omega_1 = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$ and $\Omega_2 = \{x \in \mathbb{R}^n : h(x) = 0\}$.

8.3.2.2.4. Fourth trick

Finally, if we want to minimize the function $g(x) = g(x, Ax + b) = g_1(x) + g_2(Ax + b)$, we can write:

$$\begin{aligned} \{x^{\star}, y^{\star}\} &= \arg\min_{(x,y)} g_1(x) + g_2(y) \\ &\text{s.t. } y = Ax + b \end{aligned}$$

For instance, this trick can be used for a QP problem with a nonlinear part:

$$g(x) = \frac{1}{2} x^{\top} Q x - x^{\top} R + h(x)$$

If we assume that Q is a symmetric positive-definite matrix, we set $x = Ly$, where L is the lower Cholesky matrix such that $LL^{\top} = Q$. It follows that the

ADMM form is equal to[10]:

$$\{x^\star, y^\star\} = \arg\min_{(x,y)} \underbrace{\frac{1}{2}x^\top x}_{f_x(x)} + \underbrace{h(y) - y^\top R}_{f_y(y)}$$
$$\text{s.t. } x - Ly = \mathbf{0}_n$$

We note that the x-update is straightforward because it corresponds to a standard QP problem. If we add a set Ω of constraints, we specify $f_y(y) = h(y) - y^\top R + \ _\Omega(y)$.

REMARK 8.3.– In the previous cases, we have seen that when the function $g(x)$ may contain a QP problem, it is convenient to isolate this QP problem into the x-update:

$$x^{(k+1)} = \arg\min_x \left\{ \frac{1}{2}x^\top Q x - x^\top R + \ _\Omega(x) + \frac{\varphi}{2}\left\| x - y^{(k)} + u^{(k)} \right\|_2^2 \right\}$$

Since we have:

$$\frac{\varphi}{2}\left\| x - y^{(k)} + u^{(k)} \right\|_2^2 = \frac{\varphi}{2}x^\top x - \varphi x^\top \left(y^{(k)} - u^{(k)} \right) + \frac{\varphi}{2}\left(y^{(k)} - u^{(k)} \right)^\top \left(y^{(k)} - u^{(k)} \right)$$

we deduce that the x-update is a standard QP problem where:

$$f_x^{(k+1)}(x) = \frac{1}{2}x^\top (Q + \varphi I_n) x - x^\top \left(R + \varphi \left(y^{(k)} - u^{(k)} \right) \right) + \ _\Omega(x) \quad [8.20]$$

8.3.2.3. *Application to the λ-problem of the lasso regression*

The λ-problem of the lasso regression [8.12] has the following ADMM formulation:

$$\left\{\beta^\star, \bar{\beta}^\star\right\} = \arg\min \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) + \lambda \|\bar{\beta}\|_1$$
$$\text{s.t. } \beta - \bar{\beta} = \mathbf{0}_p$$

10 This Cholesky trick has been used by Gonzalvez *et al.* (2019) to solve trend-following strategies using the ADMM algorithm in the context of Bayesian learning.

Since the x-step corresponds to a QP problem[11], we use the results given in Remark 8.3 to find the value of $\beta^{(k+1)}$:

$$\begin{aligned}\beta^{(k+1)} &= (Q+\varphi I_p)^{-1}\left(R+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\\ &= \left(X^\top X+\varphi I_p\right)^{-1}\left(X^\top Y+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\end{aligned}$$

The y-step is:

$$\begin{aligned}\bar{\beta}^{(k+1)} &= \arg\min_{\bar{\beta}}\left\{\lambda\|\bar{\beta}\|_1+\frac{\varphi}{2}\left\|\beta^{(k+1)}-\bar{\beta}+u^{(k)}\right\|_2^2\right\}\\ &= \arg\min\left\{\frac{1}{2}\left\|\bar{\beta}-\left(\beta^{(k+1)}+u^{(k)}\right)\right\|_2^2+\frac{\lambda}{\varphi}\|\bar{\beta}\|_1\right\}\end{aligned}$$

We recognize the soft-thresholding problem with $v=\beta^{(k+1)}+u^{(k)}$. Finally, the ADMM algorithm is made up of the following steps (Boyd *et al.* 2011):

$$\begin{cases}\beta^{(k+1)} = \left(X^\top X+\varphi I_p\right)^{-1}\left(X^\top Y+\varphi\left(\bar{\beta}^{(k)}-u^{(k)}\right)\right)\\ \bar{\beta}^{(k+1)} = \mathcal{S}\left(\beta^{(k+1)}+u^{(k)};\varphi^{-1}\lambda\right)\\ u^{(k+1)} = u^{(k)}+\left(\beta^{(k+1)}-\bar{\beta}^{(k+1)}\right)\end{cases}$$

8.3.3. *Proximal operators*

The x- and y-update steps of the ADMM algorithm require an ℓ_2-norm penalized optimization problem to be solved. Proximal operators are special cases of this type of problem when the matrices A or B correspond to the identity matrix I_n or its opposite $-I_n$.

8.3.3.1. *Definition*

Let $f:\mathbb{R}^n\to\mathbb{R}\cup\{+\infty\}$ be a proper closed convex function. The proximal operator $\mathbf{prox}_f(v):\mathbb{R}^n\to\mathbb{R}^n$ is defined by:

$$\mathbf{prox}_f(v)=x^\star=\arg\min_x\left\{f_v(x)=f(x)+\frac{1}{2}\|x-v\|_2^2\right\} \qquad [8.21]$$

11 We have $Q=X^\top X$ and $R=X^\top Y$.

Since the function $f_v(x) = f(x) + \frac{1}{2}\|x - v\|_2^2$ is strongly convex, it has a unique minimum for every $v \in \mathbb{R}^n$ (Parikh and Boyd 2014). By construction, the proximal operator defines a point $x^\star$ which is a trade-off between minimizing $f(x)$ and being close to v.

In many situations, we need to calculate the proximal of the scaled function $\lambda f(x)$, where $\lambda > 0$. In this case, we use the notation $\mathbf{prox}_{\lambda f}(v)$ and we have:

$$\begin{aligned}\mathbf{prox}_{\lambda f}(v) &= \arg\min_x \left\{ \lambda f(x) + \frac{1}{2}\|x - v\|_2^2 \right\} \\ &= \arg\min_x \left\{ f(x) + \frac{1}{2\lambda}\|x - v\|_2^2 \right\}\end{aligned}$$

For instance, if we consider the y-update of the ADMM algorithm with $B = -I_n$, we have:

$$\begin{aligned}y^{(k+1)} &= \arg\min_y \left\{ f_y(y) + \frac{\varphi}{2}\left\| y - v_y^{(k+1)} \right\|_2^2 \right\} \\ &= \arg\min_y \left\{ \varphi^{-1} f_y(y) + \frac{1}{2}\left\| y - v_y^{(k+1)} \right\|_2^2 \right\} \\ &= \mathbf{prox}_{\varphi^{-1} f_y}\left(v_y^{(k+1)} \right)\end{aligned}$$

where $v_y^{(k)} = Ax^{(k+1)} - c + u^{(k)}$. The interest of this mathematical formulation is to write the ADMM algorithm in a convenient form such that the x-update corresponds to the tricky part of the optimization, while the y-update is reduced to an analytical formula.

8.3.3.2. *Proximal operators and generalized projections*

In the case where $f(x) = \ _\Omega(x)$ is the indicator function, the proximal operator is then the Euclidean projection onto Ω:

$$\mathbf{prox}_f(v) = \arg\min_x \left\{ \ _\Omega(x) + \frac{1}{2}\|x - v\|_2^2 \right\}$$

$$= \arg\min_{x \in \Omega} \left\{ \|x - v\|_2^2 \right\}$$
$$= \mathcal{P}_{\Omega}(v)$$

where $\mathcal{P}_{\Omega}(v)$ is the standard projection of v onto Ω. Parikh and Boyd (2014) interpret proximal operators then as a generalization of the Euclidean projection.

Let us consider the constrained optimization problem $x^{\star} = \arg\min f(x)$ subject to $x \in \Omega$. Using the second ADMM trick, we have $f_x(x) = f(x)$, $f_y(y) = \iota_{\Omega}(y)$ and $x - y = \mathbf{0}_n$. Therefore, we can use the ADMM algorithm since the v- and y-steps become $v_y^{(k+1)} = x^{(k+1)} + u^{(k)}$ and[12] $y^{(k+1)} = \mathcal{P}_{\Omega}\left(v_y^{(k+1)}\right)$. Here, we give the results of Parikh and Boyd (2014) for some simple polyhedra:

Notation	Ω	$\mathcal{P}_{\Omega}(v)$
$\mathcal{A}_{ffineset}[A, B]$	$Ax = B$	$v - A^{\dagger}(Av - B)$
$\mathcal{H}_{yperplane}[a, b]$	$a^{\top}x = b$	$v - \dfrac{\left(a^{\top}v - b\right)}{\|a\|_2^2}a$
$\mathcal{H}_{alfspace}[c, d]$	$c^{\top}x \leq d$	$v - \dfrac{\left(c^{\top}v - d\right)_+}{\|c\|_2^2}c$
$\mathcal{B}_{ox}[x^-, x^+]$	$x^- \leq x \leq x^+$	$\mathcal{T}(v; x^-, x^+)$

where $A^{\dagger}$ is the Moore–Penrose pseudo-inverse of A and $\mathcal{T}(v; x^-, x^+)$ is the truncation operator.

8.3.3.3. *Main properties*

There are many properties that are useful for finding the analytical expression of the proximal operator. In what follows, we consider three main properties. Refer to Combettes and Pesquet (2011), Parikh and Boyd (2014) and Beck (2017) for a more exhaustive list.

---

12 We note that the parameter φ has no impact on the y-update because $\varphi^{-1} f_y(y) = f_y(y) = \iota_{\Omega}(y)$. We then deduce that:

$$\mathbf{prox}_{\varphi^{-1} f_y}\left(v_y^{(k+1)}\right) = \mathbf{prox}_{f_y}\left(v_y^{(k+1)}\right) = \mathcal{P}_{\Omega}\left(v_y^{(k+1)}\right).$$

8.3.3.3.1. Separable sum

Let us assume that $f(x) = \sum_{i=1}^{n} f_i(x_i)$ is fully separable, then the proximal of $f(v)$ is the vector of the proximal operators applied to each scalar-valued function $f_i(x_i)$:

$$\mathbf{prox}_f(v) = \begin{pmatrix} \mathbf{prox}_{f_1}(v_1) \\ \vdots \\ \mathbf{prox}_{f_n}(v_n) \end{pmatrix}$$

For example, if $f(x) = \lambda \|x\|_1$, we have $f(x) = \lambda \sum_{i=1}^{n} |x_i|$ and $f_i(x_i) = \lambda |x_i|$. We deduce that the proximal operator of $f(x)$ is the vector formulation of the soft-thresholding operator:

$$\mathbf{prox}_{\lambda\|x\|_1}(v) = \text{sign}(v) \odot (|v| - \lambda \mathbf{1}_n)_+$$

This result has been used to solve the λ-problem of the lasso regression on p. 282.

If we consider the scalar-valued logarithmic barrier function $f(x) = -\lambda \ln x$, we have:

$$f_v(x) = -\lambda \ln x + \frac{1}{2}(x - v)^2 = -\lambda \ln x + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

The first-order condition is $-\lambda x^{-1} + x - v = 0$. We obtain two roots with opposite signs: $x^\star = \frac{1}{2}\left(v \pm \sqrt{v^2 + 4\lambda}\right)$. Since the logarithmic function is defined for $x > 0$, we deduce that the proximal operator is the positive root. In the case of the vector-valued logarithmic barrier $f(x) = -\lambda \sum_{i=1}^{n} \ln x_i$, it follows that:

$$\mathbf{prox}_f(v) = \frac{v + \sqrt{v \odot v + 4\lambda}}{2}$$

8.3.3.3.2. Moreau decomposition

An important property of the proximal operator is the Moreau decomposition theorem:

$$\mathbf{prox}_f(v) + \mathbf{prox}_{f^*}(v) = v$$

where f^* is the convex conjugate of f. This result is used extensively to find the proximal of norms, the max function, the sum-of-k-largest-values function, etc. (Beck 2017).

In the case of the pointwise maximum function $f(x) = \max x$, we can show that:

$$\mathbf{prox}_{\lambda \max x}(v) = \min(v, s^\star)$$

where $s^\star$ is the solution of the following equation:

$$s^\star = \left\{ s \in \mathbb{R} : \sum_{i=1}^{n} (v_i - s)_+ = \lambda \right\}$$

If we assume that $f(x) = \|x\|_p$, we obtain:

p	$\mathbf{prox}_{\lambda f}(v)$
$p = 1$	$\mathcal{S}(v; \lambda) = \text{sign}(v) \odot (\lvert v \rvert - \lambda \mathbf{1}_n)_+$
$p = 2$	$\left(1 - \frac{\lambda}{\max(\lambda, \|v\|_2)}\right) v$
$p = \infty$	$\text{sign}(v) \odot \mathbf{prox}_{\lambda \max x}(\lvert v \rvert)$

If $f(x)$ is an $\boldsymbol{\ell}_q$-norm function, then $f^*(x) = \quad_{\mathcal{B}_p}(x)$, where $\mathcal{B}_p$ is the $\boldsymbol{\ell}_p$ unit ball and $p^{-1} + q^{-1} = 1$. Since we have $\mathbf{prox}_{f^*}(v) = \mathcal{P}_{\mathcal{B}_p}(v)$, we deduce that:

$$\mathbf{prox}_f(v) + \mathcal{P}_{\mathcal{B}_p}(v) = v$$

More generally, we have:

$$\mathbf{prox}_{\lambda f}(v) + \lambda \mathcal{P}_{\mathcal{B}_p}\left(\frac{v}{\lambda}\right) = v$$

It follows that the projection onto the $\boldsymbol{\ell}_p$ ball can be deduced from the proximal operator of the $\boldsymbol{\ell}_q$-norm function. Let $\mathcal{B}_p(c,\lambda) = \{x \in \mathbb{R}^n : \|x-c\|_p \leq \lambda\}$ be the $\boldsymbol{\ell}_p$ ball with center c and radius λ. We obtain:

p	$\mathcal{P}_{\mathcal{B}_p(\mathbf{0}_n,\lambda)}(v)$	q
$p=1$	$v - \text{sign}(v) \odot \mathbf{prox}_{\lambda \max x}(\lvert v\rvert)$	$q=\infty$
$p=2$	$v - \mathbf{prox}_{\lambda\|x\|_2}(v)$	$q=2$
$p=\infty$	$\mathcal{T}(v;-\lambda,\lambda)$	$q=1$

8.3.3.3.3. Scaling and translation

Let us define $g(x) = f(ax+b)$, where $a \neq 0$. We have[13]:

$$\mathbf{prox}_g(v) = \frac{\mathbf{prox}_{a^2 f}(av+b) - b}{a}$$

We can use this property when the center c of the $\boldsymbol{\ell}_p$ ball is not equal to $\mathbf{0}_n$. Since we have $\mathbf{prox}_g(v) = \mathbf{prox}_f(v-c) + c$, where $g(x) = f(x-c)$, and the equivalence $\mathcal{B}_p(\mathbf{0}_n,\lambda) = \{x \in \mathbb{R}^n : f(x) \leq \lambda\}$, where $f(x) = \|x\|_p$, we deduce that:

$$\mathcal{P}_{\mathcal{B}_p(c,\lambda)}(v) = \mathcal{P}_{\mathcal{B}_p(\mathbf{0}_n,\lambda)}(v-c) + c$$

8.3.3.4. *Application to the τ-problem of the lasso regression*

We have previously presented the lasso regression problem by considering the Lagrange formulation (λ-problem). We now consider the original τ-problem:

$$\hat{\beta}(\tau) = \arg\min_{\beta} \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) \qquad \text{s.t.} \quad \|\beta\|_1 \leq \tau$$

The ADMM formulation is:

$$\begin{aligned}\{\beta^\star, \bar{\beta}^\star\} &= \arg\min_{(\beta,\bar{\beta})} \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) + \Omega(\bar{\beta}) \\ &\text{s.t. } \beta = \bar{\beta}\end{aligned}$$

13 The proof can be found in Beck (2017, p. 138).

where $\Omega = \mathcal{B}_1(\mathbf{0}_n, \tau)$ is the centered ℓ_1 ball with radius τ. We note that the x-update is:

$$\begin{aligned}\beta^{(k+1)} &= \arg\min_{\beta} \left\{ \frac{1}{2}(Y - X\beta)^\top (Y - X\beta) + \frac{\varphi}{2} \left\| \beta - \bar{\beta}^{(k)} + u^{(k)} \right\|_2^2 \right\} \\ &= \left(X^\top X + \varphi I_p \right)^{-1} \left(X^\top Y + \varphi \left(\bar{\beta}^{(k)} - u^{(k)} \right) \right)\end{aligned}$$

where $v_x^{(k+1)} = \bar{\beta}^{(k)} - u^{(k)}$. For the y-update, we deduce that:

$$\begin{aligned}\bar{\beta}^{(k+1)} &= \arg\min_{\bar{\beta}} \left\{ \Omega\left(\bar{\beta}\right) + \frac{\varphi}{2} \left\| \beta^{(k+1)} - \bar{\beta} + u^{(k)} \right\|_2^2 \right\} \\ &= \mathcal{P}_\Omega\left(v_y^{(k+1)} \right) \\ &= v_y^{(k+1)} - \operatorname{sign}\left(v_y^{(k+1)} \right) \odot \mathbf{prox}_{\tau \max x}\left(\left\| v_y^{(k+1)} \right\| \right)\end{aligned}$$

where $v_y^{(k+1)} = \beta^{(k+1)} + u^{(k)}$. Finally, the u-update is defined by $u^{(k+1)} = u^{(k)} + \beta^{(k+1)} - \bar{\beta}^{(k+1)}$.

REMARK 8.4.– The ADMM algorithm is similar for λ- and τ-problems since the only difference concerns the y-step. For the λ-problem, we apply the soft-thresholding operator while we use the ℓ_1 projection in the case of the τ-problem. However, our experience shows that the τ-problem is easier to solve with the ADMM algorithm from a practical point of view. The reason is that the y-update of the τ-problem is independent of the penalization parameter φ. This is not the case for the λ-problem, because the soft-thresholding depends on the value taken by $\varphi^{-1}\lambda$.

8.3.4. *Dykstra's algorithm*

We now consider the proximal optimization problem where the function $f(x)$ is the convex sum of basic functions $f_j(x)$:

$$x^\star = \arg\min_x \left\{ \sum_{j=1}^m f_j(x) + \frac{1}{2}\|x - v\|_2^2 \right\}$$

and the proximal of each basic function is known.

8.3.4.1. *The $m = 2$ case*

In the previous section, we listed some analytical solutions of the proximal problem when the function $f(x)$ is basic. For instance, we know the proximal solution of the ℓ_1-norm function $f_1(x) = \lambda_1 \|x\|_1$ or the proximal solution of the logarithmic barrier function $f_2(x) = \lambda_2 \sum_{i=1}^{n} \ln x_i$. However, we do not know how to compute the proximal operator of $f(x) = f_1(x) + f_2(x)$:

$$x^\star = \mathbf{prox}_f(v) = \arg\min_x f_1(x) + f_2(x) + \frac{1}{2}\|x - v\|_2^2$$

Nevertheless, an elegant solution is provided by Dykstra's algorithm (Dykstra 1983; Bauschke and Borwein 1994; Combettes and Pesquet 2011), which is defined by the following iterations:

$$\begin{cases} x^{(k+1)} = \mathbf{prox}_{f_1}\left(y^{(k)} + p^{(k)}\right) \\ p^{(k+1)} = y^{(k)} + p^{(k)} - x^{(k+1)} \\ y^{(k+1)} = \mathbf{prox}_{f_2}\left(x^{(k+1)} + q^{(k)}\right) \\ q^{(k+1)} = x^{(k+1)} + q^{(k)} - y^{(k+1)} \end{cases} \quad [8.22]$$

where $x^{(0)} = y^{(0)} = v$ and $p^{(0)} = q^{(0)} = \mathbf{0}_n$. This algorithm is obviously related to the Douglas–Rachford splitting framework[14], where $x^{(k)}$ and $p^{(k)}$ are the variable and the residual associated with $f_1(x)$, and $y^{(k)}$ and $q^{(k)}$ are the variable and the residual associated with $f_2(x)$, respectively.

8.3.4.2. *The $m > 2$ case*

The case $m > 2$ is a generalization of the previous algorithm by considering m residuals:

1) the x-update is:

$$x^{(k+1)} = \mathbf{prox}_{f_{j(k)}}\left(x^{(k)} + z^{(k+1-m)}\right)$$

2) the z-update is:

$$z^{(k+1)} = x^{(k)} + z^{(k+1-m)} - x^{(k+1)}$$

14 See Combettes and Pesquet (2011).

where $x^{(0)} = v$, $z^{(k)} = \mathbf{0}_n$ for $k < 0$ and $j(k) = \text{mod}(k+1, m)$ denotes the modulo operator taking values in $\{1, \ldots, m\}$. The variable $x^{(k)}$ is updated at each iteration, while the residual $z^{(k)}$ is updated every m iterations. This implies that the basic function $f_j(x)$ is related to the residuals $z^{(j)}$, $z^{(j+m)}$, $z^{(j+2m)}$, etc. Following Tibshirani (2017), it is better to write Dykstra's algorithm by using two iteration indices k and j. The main index k refers to the cycle[15], whereas the sub-index j refers to the constraint number:

1) The x-update is:

$$x^{(k+1,j)} = \mathbf{prox}_{f_j}\left(x^{(k+1,j-1)} + z^{(k,j)}\right) \quad [8.23]$$

2) The z-update is:

$$z^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - x^{(k+1,j)} \quad [8.24]$$

where $x^{(1,0)} = v$, $z^{(k,j)} = \mathbf{0}_n$ for $k = 0$ and $x^{(k+1,0)} = x^{(k,m)}$.

Dykstra's algorithm is particularly efficient when we consider the projection problem $x^\star = \mathcal{P}_\Omega(v)$, where $\Omega = \Omega_1 \cap \Omega_2 \cap \cdots \cap \Omega_m$. Indeed, the solution is found by replacing equation [8.23] with:

$$x^{(k+1,j)} = \mathcal{P}_{\Omega_j}\left(x^{(k+1,j-1)} + z^{(k,j)}\right) \quad [8.25]$$

8.3.4.3. *Application to general linear constraints*

Let us consider the case $\Omega = \{x \in \mathbb{R}^n : Cx \leq D\}$ where the number of inequality constraints is equal to m. We can write $\Omega = \Omega_1 \cap \Omega_2 \cap \cdots \cap \Omega_m$, where $\Omega_j = \left\{x \in \mathbb{R}^n : c_{(j)}^\top x \leq d_{(j)}\right\}$, $c_{(j)}^\top$ corresponds to the j^{th} row of C and $d_{(j)}$ is the j^{th} element of D. Since the projection $\mathcal{P}_{\Omega_j}$ is known and has been given on p. 101, we can find the projection $\mathcal{P}_\Omega$ using Algorithm [8.2.]. If we consider $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$, we decompose Ω as the intersection of three basic convex sets $\Omega = \Omega_1 \cap \Omega_2 \cap \Omega_3$, where $\Omega_1 = \{x \in \mathbb{R}^n : Ax = B\}$, $\Omega_2 = \{x \in \mathbb{R}^n : Cx \leq D\}$ and $\Omega_3 = \{x \in \mathbb{R}^n : x^- \leq x \leq x^+\}$. Using Dykstra's algorithm is equivalent to formulating Algorithm [8.3.].

15 Exactly like the coordinate descent algorithm.

Algorithm 8.2. Dykstra's algorithm for solving the proximal problem with linear inequality constraints

The goal is to compute the solution $x^\star = \mathbf{prox}_f(v)$ where $f(x) = \quad_\Omega(x)$ and $\Omega = \{x \in \mathbb{R}^n : Cx \leq D\}$
We initialize $x^{(0,m)} \leftarrow v$
We set $z^{(0,1)} \leftarrow \mathbf{0}_n, \ldots, z^{(0,m)} \leftarrow \mathbf{0}_n$
$k \leftarrow 0$
repeat
  $x^{(k+1,0)} \leftarrow x^{(k,m)}$
  for $j = 1 : m$ **do**
    The x-update is:

$$x^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - \frac{\left(c_{(j)}^\top x^{(k+1;j-1)} + c_{(j)}^\top z^{(k,j)} - d_{(j)}\right)_+}{\left\|c_{(j)}\right\|_2^2} c_{(j)}$$

    The z-update is:

$$z^{(k+1,j)} = x^{(k+1,j-1)} + z^{(k,j)} - x^{(k+1,j)}$$

  end for
  $k \leftarrow k + 1$
until Convergence
return $x^\star \leftarrow x^{(k,m)}$

Since we have:

$$\frac{1}{2}\|x - v\|_2^2 = \frac{1}{2}x^\top x - x^\top v + \frac{1}{2}v^\top v$$

we deduce that the two previous problems can be cast into a QP problem:

$$x^\star = \arg\min_x \frac{1}{2}x^\top I_n x - x^\top v \qquad \text{s.t.} \quad x \in \Omega$$

We can then compare the efficiency of Dykstra's algorithm with the QP algorithm. Let us consider the proximal problem where the vector v is defined by the elements $v_i = \ln\left(1 + i^2\right)$ and the set of constraints is:

$$\Omega = \left\{x \in \mathbb{R}^n : \sum_{i=1}^n x_i \leq \frac{1}{2}, \sum_{i=1}^n e^{-i} x_i \geq 0\right\}$$

Algorithm 8.3. Dykstra's algorithm for solving the proximal problem with general linear constraints

The goal is to compute the solution $x^\star = \mathbf{prox}_f(v)$ where $f(x) = \quad_\Omega(x)$ and $\Omega = \{x \in \mathbb{R}^n : Ax = B, Cx \leq D, x^- \leq x \leq x^+\}$
We initialize $x_m^{(0)} \leftarrow v$
We set $z_1^{(0)} \leftarrow \mathbf{0}_n$, $z_2^{(0)} \leftarrow \mathbf{0}_n$ and $z_3^{(0)} \leftarrow \mathbf{0}_n$
$k \leftarrow 0$
repeat
$x_0^{(k+1)} \leftarrow x_m^{(k)}$
$x_1^{(k+1)} \leftarrow x_0^{(k+1)} + z_1^{(k)} - A^\dagger\left(Ax_0^{(k+1)} + Az_1^{(k)} - B\right)$
$z_1^{(k+1)} \leftarrow x_0^{(k+1)} + z_1^{(k)} - x_1^{(k+1)}$
$x_2^{(k+1)} \leftarrow \mathcal{P}_{\Omega_2}\left(x_1^{(k+1)} + z_2^{(k)}\right)$ ▶ Algorithm (8.2.)
$z_2^{(k+1)} \leftarrow x_1^{(k+1)} + z_2^{(k)} - x_2^{(k+1)}$
$x_3^{(k+1)} \leftarrow \mathcal{T}\left(x_2^{(k+1)} + z_3^{(k)}; x^-, x^+\right)$
$z_3^{(k+1)} \leftarrow x_2^{(k+1)} + z_3^{(k)} - x_3^{(k+1)}$
$k \leftarrow k + 1$
until Convergence
return $x^\star \leftarrow x_3^{(k)}$

Using a MATLAB implementation[16], we find that the computational time of Dykstra's algorithm, when n is equal to 10 million, is equal to the QP algorithm when n is equal to 12,500, meaning that there is a factor of 800 between the two methods!

8.4. Applications to portfolio optimization

The development of the previous algorithms will fundamentally change the practice of portfolio optimization. Until now, we have seen that portfolio managers live in a quadratic programming world. With these new optimization algorithms, we can consider more complex portfolio optimization programs with non-quadratic objective function, regularization with penalty functions and nonlinear constraints.

16 The QP implementation corresponds to the `quadprog` function.

Item	Portfolio	$f(x)$	Reference
(1)	MVO	$\frac{1}{2}x^\top \Sigma x - \gamma x^\top \mu$	Markowitz (1952)
(2)	GMV	$\frac{1}{2}x^\top \Sigma x$	Jagganathan and Ma (2003)
(3)	MDP	$\ln\left(\sqrt{x^\top \Sigma x}\right) - \ln\left(x^\top \sigma\right)$	Choueifaty and Coignard (2008)
(4)	KL	$\sum_{i=1}^n x_i \ln\left(x_i / \tilde{x}_i\right)$	Bera and Park (2008)
(5)	ERC	$\frac{1}{2}x^\top \Sigma x - \lambda \sum_{i=1}^n \ln x_i$	Maillard *et al.* (2010)
(6)	RB	$\mathcal{R}(x) - \lambda \sum_{i=1}^n \mathcal{RB}_i \cdot \ln x_i$	Roncalli (2015)
(7)	RQE	$\frac{1}{2}x^\top D x$	Carmichael *et al.* (2018)

Table 8.1. *Some objective functions used in portfolio optimization*

We consider a universe of n assets. Let x be the vector of weights in the portfolio. We denote by μ and Σ the vector of expected returns and the covariance matrix of asset returns[17]. Some models also consider a reference portfolio $\tilde{x}$. In Table 8.1, we report the main objective functions that are used by professionals[18]. Besides the mean-variance optimized portfolio (MVO) and the global minimum variance portfolio (GMV), we find the equal risk contribution portfolio (ERC), the risk budgeting portfolio (RB) and the most diversified portfolio (MDP). According to Choueifaty and Coignard (2008), the MDP is defined as the portfolio which maximizes the diversification ratio $\mathcal{DR}(x) = \dfrac{x^\top \sigma}{\sqrt{x^\top \Sigma x}}$. We also include in the list two "academic" portfolios, which are based on the Kullback–Leibler (KL) information criteria and the Rao's quadratic entropy (RQE) measure[19].

In a similar way, we list in Table 8.2 some popular regularization penalty functions that are used in the industry (Bruder *et al.* 2013; Bourgeron *et al.* 2018). The ridge and lasso regularization are well known in statistics and machine learning (Hastie *et al.* 2009). The log-barrier penalty function comes from the risk budgeting optimization problem, whereas Shannon's entropy is another approach for imposing a sufficient weight diversification.

17 The vector of volatilities is defined by $\sigma = (\sigma_1, \ldots, \sigma_n)$.

18 For each model, we write the optimization problem as a minimization problem.

19 D is the dissimilarity matrix satisfying $D_{i,j} \geq 0$ and $D_{i,j} = D_{j,i}$.

Item	Regularization	$\mathfrak{R}(x)$	Reference
(8)	Ridge	$\lambda \|x - \tilde{x}\|_2^2$	DeMiguel *et al.* (2009)
(9)	Lasso	$\lambda \|x - \tilde{x}\|_1$	Brodie *at al.* (2009)
(10)	Log-barrier	$-\sum_{i=1}^{n} \lambda_i \ln x_i$	Roncalli (2013)
(11)	Shannon's entropy	$\lambda \sum_{i=1}^{n} x_i \ln x_i$	Yu *et al.* (2014)

Table 8.2. *Some regularization penalties used in portfolio optimization*

Concerning the constraints, the most famous are the no cash/no leverage and no short selling restrictions. Weight bounds and asset class limits are also extensively used by practitioners. Turnover and transaction cost management may be an important topic when rebalancing a current portfolio $\tilde{x}$. When managing long/short portfolios, we generally impose leverage or long/short exposure limits. In the case of a benchmarked strategy, we might also want to have a tracking error limit, with respect to the benchmark $\tilde{x}$. On the contrary, we can impose a minimum tracking error or active share in the case of active management. Finally, the Herfindahl constraint is used for some smart beta portfolios.

In what follows, we consider several portfolio optimization problems. Most of them are a combination of an objective function, one or two regularization penalty functions and some constraints that have been listed above. From an industrial point of view, it is interesting to implement the proximal operator for each item. In this approach, solving any portfolio optimization problem is equivalent to using CCD, ADMM, Dykstra and the appropriate proximal functions as Lego bricks.

8.4.1. *Minimum variance optimization*

8.4.1.1. *Managing diversification*

The global minimum variance (GMV) portfolio corresponds to the following optimization program:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \qquad \text{s.t.} \quad \mathbf{1}_n^\top x = 1$$

Item	Constraint	Formula
(12)	No cash and leverage	$\sum_{i=1}^{n} x_i = 1$
(13)	No short selling	$x_i \geq 0$
(14)	Weight bounds	$x_i^- \leq x_i \leq x_i^+$
(15)	Asset class limits	$c_j^- \leq \sum_{i \in \mathcal{C}_j} x_i \leq c_j^+$
(16)	Turnover	$\sum_{i=1}^{n} \lvert x_i - \tilde{x}_i \rvert \leq \boldsymbol{\tau}^+$
(17)	Transaction costs	$\sum_{i=1}^{n} \left(c_i^- \left(\tilde{x}_i - x_i \right)_+ + c_i^+ \left(x_i - \tilde{x}_i \right)_+ \right) \leq \boldsymbol{c}^+$
(18)	Leverage limit	$\sum_{i=1}^{n} \lvert x_i \rvert \leq \mathcal{L}^+$
(19)	Long/short exposure	$-\mathcal{LS}^- \leq \sum_{i=1}^{n} x_i \leq \mathcal{LS}^+$
(20)	Benchmarking	$\sqrt{(x - \tilde{x})^\top \Sigma (x - \tilde{x})} \leq \sigma^+$
(21)	Tracking error floor	$\sqrt{(x - \tilde{x})^\top \Sigma (x - \tilde{x})} \geq \sigma^-$
(22)	Active share floor	$\frac{1}{2} \sum_{i=1}^{n} \lvert x_i - \tilde{x}_i \rvert \geq \mathcal{AS}^-$
(23)	Number of active bets	$\left(x^\top x \right)^{-1} \geq \mathcal{N}^-$

Table 8.3. *Some constraints used in portfolio optimization*

We know that the solution is $x^\star = \left(\mathbf{1}_n^\top \Sigma^{-1} \mathbf{1}_n \right)^{-1} \Sigma^{-1} \mathbf{1}_n$. In practice, nobody implements the GMV portfolio because it is a long/short portfolio and it is not robust. Most of the time, professionals impose weight bounds: $0 \leq x_i \leq x^+$. However, this approach generally leads to corner solutions, meaning that a large number of optimized weights are equal to zero or the upper bound and very few assets have a weight within the range. With the emergence of smart beta portfolios, the minimum variance portfolio gained popularity among institutional investors. For instance, we can find many passive indices based on this framework. In order to increase the robustness of these portfolios, the first generation of minimum variance strategies has used relative weight bounds with respect to a benchmark b: $\delta^- b_i \leq x_i \leq \delta^+ b_i$, where $0 < \delta^- < 1$ and $\delta^+ > 1$. For instance, the most popular scheme is to take $\delta^- = 0.5$ and $\delta^+ = 2$. Nevertheless, this constraint produces the illusion that the portfolio is diversified, because the optimized weights are different. In fact, portfolio weights are different because benchmark weights are different. The second generation of minimum variance strategies imposes a global diversification constraint. The most popular solution is based on the Herfindahl index $\mathcal{H}(x) = \sum_{i=1}^{n} x_i^2$. This index takes the value 1 for a pure concentrated portfolio ($\exists\, i : x_i = 1$) and $1/n$ for an equally weighted portfolio. Therefore, we can define the number of effective bets as the inverse of the Herfindahl index: $\mathcal{N}(x) = \mathcal{H}(x)^{-1}$.

The optimization program becomes:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \quad [8.26]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ \mathcal{N}(x) \geq \mathcal{N}^- \end{cases}$$

where $\mathcal{N}^-$ is the minimum number of effective bets. The Herfindhal constraint is equivalent to:

$$\mathcal{N}(x) \geq \mathcal{N}^- \Leftrightarrow \left(x^\top x\right)^{-1} \geq \mathcal{N}^- \Leftrightarrow x^\top x \leq \frac{1}{\mathcal{N}^-}$$

Therefore, a first solution to solve [8.26] is to consider the following QP problem[20]:

$$x^\star(\lambda) = \arg\min_x \frac{1}{2} x^\top \Sigma x + \lambda x^\top x \quad [8.27]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \end{cases}$$

where $\lambda \geq 0$ is a scalar. Since $\mathcal{N}(x^\star(\infty))$ is equal to the number n of assets and $\mathcal{N}(x^\star(\lambda))$ is an increasing function of λ, problem [8.27] has a unique solution if $\mathcal{N}^- \in [\mathcal{N}(x^\star(0)), n]$. There is an optimal value $\lambda^\star$ such that for each $\lambda \geq \lambda^\star$, we have $\mathcal{N}(x^\star(\lambda)) \geq \mathcal{N}^-$. Computing the optimal portfolio $x^\star(\lambda^\star)$ therefore implies finding the solution $\lambda^\star$ of the nonlinear equation $\mathcal{N}(x^\star(\lambda)) = \mathcal{N}^-$.

A second method is to consider the ADMM form:

$$\{x^\star, y^\star\} = \arg\min_{(x,y)} \frac{1}{2} x^\top \Sigma x + \Omega_3(x) + \Omega_4(y)$$
$$\text{s.t. } x = y$$

20 The objective function can be written as:

$$\frac{1}{2} x^\top \Sigma x + \lambda x^\top x = \frac{1}{2} x^\top (\Sigma + 2I_n) x.$$

where $\Omega_3 = \mathcal{H}_{yperplane}\left[\mathbf{1}_n, 1\right]$ and $\Omega_4 = \mathcal{B}_{ox}\left[\mathbf{0}_n, x^+\right] \cap \mathcal{B}_2\left(\mathbf{0}_n, \sqrt{\frac{1}{\mathcal{N}^-}}\right)$. In this case, the x- and y-updates become[21]:

$$\begin{aligned} x^{(k+1)} &= \arg\min_x \left\{ \frac{1}{2} x^\top \left(\Sigma + \varphi I_n\right) x - \varphi x^\top \left(y^{(k)} - u^{(k)}\right) + \quad _{\Omega_3}(x) \right\} \\ &= \left(\Sigma + \varphi I_n\right)^{-1} \left(\varphi \left(y^{(k)} - u^{(k)}\right) \right. \\ &\quad \left. + \frac{1 - \mathbf{1}_n^\top \left(\Sigma + \varphi I_n\right)^{-1} \varphi \left(y^{(k)} - u^{(k)}\right)}{\mathbf{1}_n^\top \left(\Sigma + \varphi I_n\right)^{-1} \mathbf{1}_n} \mathbf{1}_n \right) \end{aligned}$$

and:

$$y^{(k+1)} = \mathcal{P}_{\mathcal{B}\text{ox}-\mathcal{B}\text{all}}\left(x^{(k+1)} + u^{(k)}; \mathbf{0}_n, x^+, \mathbf{0}_n, \sqrt{\frac{1}{\mathcal{N}^-}}\right)$$

where $\mathcal{P}_{\mathcal{B}\text{ox}-\mathcal{B}\text{all}}$ corresponds to Dykstra's algorithm given in Appendix 8.7.1.4.1.

We consider the parameter set #1 given in Appendix 8.7.2.1. The investment universe is made up of eight stocks. We would like to build a diversified minimum variance long-only portfolio without imposing an upper weight bound[22]. In Table 8.4, we report the solutions found by the ADMM algorithm for several values of $\mathcal{N}^-$. When there is no Herfindahl constraint, the portfolio is fully invested in the seventh stock, meaning that the asset diversification is very poor. Then, we increase the number of effective bets. If $\mathcal{N}^-$ is equal to the number n of stocks, we verify that the solution corresponds to the equally weighted portfolio. Between these two limit cases, we see the impact of the Herfindahl constraint on the portfolio diversification. The parameter set #1 is defined with respect to a capitalization-weighted index, whose weights are equal to 23%, 19%, 17%, 9%, 8%, 6% and 5%. The number of effective bets of this benchmark is equal to 6.435. If we impose that the effective number of bets of the minimum variance portfolio is at least equal to the effective number of bets of the benchmark, we find the following solution: 14.74%, 15.45%, 1.79%, 15.49%, 6.17%, 13.83%, 23.21% and 9.31%.

21 See Appendix 8.7.1.2 for the derivation of the x-update.

22 This means that x^+ is set to $\mathbf{1}_n$.

$\mathcal{N}^-$	1.00	2.00	3.00	4.00	5.00	6.00	6.50	7.00	7.50	8.00
$x_1^\star$	0.00	3.22	9.60	13.83	15.18	15.05	14.69	14.27	13.75	12.50
$x_2^\star$	0.00	12.75	14.14	15.85	16.19	15.89	15.39	14.82	14.13	12.50
$x_3^\star$	0.00	0.00	0.00	0.00	0.00	0.07	2.05	4.21	6.79	12.50
$x_4^\star$	0.00	10.13	15.01	17.38	17.21	16.09	15.40	14.72	13.97	12.50
$x_5^\star$	0.00	0.00	0.00	0.00	0.71	5.10	6.33	7.64	9.17	12.50
$x_6^\star$	0.00	5.36	8.95	12.42	13.68	14.01	13.80	13.56	13.25	12.50
$x_7^\star$	100.00	68.53	52.31	40.01	31.52	25.13	22.92	20.63	18.00	12.50
$x_8^\star$	0.00	0.00	0.00	0.50	5.51	8.66	9.41	10.14	10.95	12.50

Table 8.4. *Minimum variance portfolios (in %)*

As explained previously, we can also solve the optimization problem by combining problem [8.27] and the bisection algorithm. However, this approach is no longer valid if we consider diversification constraints that are not quadratic. For instance, let us consider the generalized minimum variance problem:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x \qquad [8.28]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ \mathcal{D}(x) \geq \mathcal{D}^- \end{cases}$$

where $\mathcal{D}(x)$ is a weight diversification measure and $\mathcal{D}^-$ is the minimum acceptable diversification. For example, we can use Shannon's entropy, the Gini index or the diversification ratio. In this case, it is not possible to obtain an equivalent QP problem, whereas the ADMM algorithm is exactly the same as previously, except for the y-update:

$$y^{(k+1)} = \mathcal{P}_{\mathcal{B}_{ox}[\mathbf{0}_n, x^+] \cap \mathfrak{D}} \left(x^{(k+1)} + u^{(k)} \right)$$

where $\mathfrak{D} = \{x \in \mathbb{R}^n : \mathcal{D}(x) \geq \mathcal{D}^-\}$. The projection onto $\mathfrak{D}$ can be easily derived from the proximal operator of the dual function (see section 8.4.4).

REMARK 8.5.– If we compare the computational times, we observe that the best method is the ADMM algorithm. In our example, the computational time is divided by a factor of eight with respect to the bisection approach. If we consider a large-scale problem with n larger than 1,000, the computational time is generally divided by a factor greater than 50!

8.4.1.2. *Managing the portfolio rebalancing process*

Another big challenge of the minimum variance portfolio is the management of the turnover between two rebalancing dates. Let x_t be the current portfolio. The optimization program for calibrating the optimal solution x_{t+1} for the next rebalancing date $t+1$ may include a penalty function $\boldsymbol{c}(x \mid x_t)$ and/or a weight constraint $\mathfrak{C}(x \mid x_t)$ that are parameterized with respect to the current portfolio x_t:

$$x_{t+1} = \arg\min_x \frac{1}{2} x^\top \Sigma x + \boldsymbol{c}(x \mid x_t) \quad [8.29]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq x^+ \\ x \in \mathfrak{C}(x \mid x_t) \end{cases}$$

Again, we can solve this problem using the ADMM algorithm. Thanks to Dykstra's algorithm, the only difficulty is finding the proximal operator of $\boldsymbol{c}(x \mid x_t)$ or $\mathfrak{C}(x \mid x_t)$ when performing the y-update.

Let us define the cost function as:

$$\boldsymbol{c}(x \mid x_t) = \lambda \sum_{i=1}^n \left(c_i^- (x_{i,t} - x_i)_+ + c_i^+ (x_i - x_{i,t})_+ \right)$$

where c_i^- and c_i^+ are the bid and ask transaction costs. In Appendix 8.7.1.6, we show that the proximal operator is:

$$\mathbf{prox}_{\boldsymbol{c}(x|x_t)}(v) = x_t + \mathcal{S}\left(v - x_t; \lambda c^-, \lambda c^+\right) \quad [8.30]$$

where $\mathcal{S}(v; \lambda_-, \lambda_+) = (v - \lambda_+)_+ - (v + \lambda_-)_-$ is the two-sided soft-thresholding operator.

If we define the cost constraint $\mathfrak{C}(x \mid x_t)$ as a turnover constraint:

$$\mathfrak{C}(x \mid x_t) = \left\{x \in \mathbb{R}^n : \|x - x_t\|_1 \leq \boldsymbol{\tau}^+\right\}$$

the proximal operator is:

$$\mathcal{P}_{\mathfrak{C}}(v) = v - \operatorname{sign}(v - x_t) \odot \min\left(|v - x_t|, s^\star\right) \quad [8.31]$$

where $s^\star = \left\{s \in \mathbb{R} : \sum_{i=1}^n \left(|v_i - x_{t,i}| - s\right)_+ = \boldsymbol{\tau}^+\right\}$.

REMARK 8.6.– These two examples are very basic and show how we can easily introduce turnover management using the ADMM framework. More sophisticated approaches are presented in section 8.4.4.

8.4.2. *Smart beta portfolios*

In this section, we consider three main models of smart beta portfolios: the equal risk contribution (ERC) portfolio, the risk budgeting (RB) portfolio and the most diversified portfolio (MDP). Specific algorithms for these portfolios based on the CCD method have already been presented in Griveau-Billion *et al.* (2013) and Richard and Roncalli (2015, 2019). We extend these results to the ADMM algorithm.

8.4.2.1. *The ERC portfolio*

The ERC portfolio uses the volatility risk measure $\sigma(x) = \sqrt{x^\top \Sigma x}$ and allocates the weights such that the risk contribution is the same for all the assets of the portfolio (Maillard *et al.* 2010):

$$\mathcal{RC}_i(x) = x_i \frac{\partial \sigma(x)}{\partial x_i} = x_j \frac{\partial \sigma(x)}{\partial x_j} = \mathcal{RC}_j(x)$$

In this case, we can show that the ERC portfolio is the scaled solution $x^\star / \left(\mathbf{1}_n^\top x^\star\right)$, where $x^\star$ is given by:

$$x^\star = \arg\min_x \frac{1}{2} x^\top \Sigma x - \lambda \sum_{i=1}^n \ln x_i$$

and λ is any positive scalar. The first-order condition is $(\Sigma x)_i - \lambda x_i^{-1} = 0$. It follows that $x_i (\Sigma x)_i - \lambda = 0$ or $x_i^2 \sigma_i^2 + x_i \sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j - \lambda = 0$.

We deduce that the solution is the positive root of the second-degree equation. Finally, we obtain the following iteration for the CCD algorithm:

$$x_i^{(k+1)} = \frac{-v_i^{(k+1)} + \sqrt{\left(v_i^{(k+1)}\right)^2 + 4\lambda\sigma_i^2}}{2\sigma_i^2} \quad [8.32]$$

where:

$$v_i^{(k+1)} = \sigma_i \sum_{j<i} x_j^{(k+1)} \rho_{i,j}\sigma_j + \sigma_i \sum_{j>i} x_j^{(k)} \rho_{i,j}\sigma_j$$

The ADMM algorithm uses the first trick where $f_x(x) = \frac{1}{2}x^\top \Sigma x$ and $f_y(y) = -\lambda \sum_{i=1}^n \ln y_i$. It follows that the x- and y-update steps are:

$$x^{(k+1)} = (\Sigma + \varphi I_n)^{-1} \varphi \left(y^{(k)} - u^{(k)}\right)$$

and:

$$y_i^{(k+1)} = \frac{1}{2}\left(\left(x_i^{(k+1)} + u_i^{(k)}\right) + \sqrt{\left(x_i^{(k+1)} + u_i^{(k)}\right)^2 + 4\lambda\varphi^{-1}}\right)$$

We apply the CCD and ADMM algorithms to the parameter set #1. We find that the ERC portfolio is equal to 11.40%, 12.29%, 5.49%, 11.91%, 6.65%, 10.81%, 33.52% and 7.93%. It appears that the CCD algorithm is much more efficient than the ADMM algorithm. For instance, if we set $\lambda = \sqrt{x^{(0)\top}\Sigma x^{(0)}}$, $x^{(0)} = n^{-1}\mathbf{1}_n$ and $\varphi = 1$, the CCD algorithm needs six cycles to converge, whereas the ADMM algorithm needs 156 iterations if we set the convergence criterion[23] $\varepsilon = 10^{-8}$. Whatever the values of λ, $x^{(0)}$ and ε, our experience is that the CCD generally converges within less than 15 cycles even if the number of assets is greater than 1,000. The convergence of the ADMM is more of an issue because it depends on the parameters λ and φ. In Figure 8.2, we have reported the number of iterations of the ADMM with respect to φ for several values of ε when $\lambda = 1$ and $x^{(0)} = \mathbf{1}_n$. We verify that it is very sensitive to the value taken by φ. Curiously, the parameter λ has little influence, meaning that the convergence issue mainly concerns the x-update step.

23 The termination rule is defined as $\max \left|x_i^{(k+1)} - x_i^{(k)}\right| \leq \varepsilon$.

8.4.2.2. *Risk budgeting optimization*

The ERC portfolio has been extended by Roncalli (2013) when the risk budgets are not equal and when the risk measure $\mathcal{R}(x)$ is convex and coherent:

$$\mathcal{RC}_i(x) = x_i \frac{\partial \mathcal{R}(x)}{\partial x_i} = \mathcal{RB}_i$$

where $\mathcal{RB}_i$ is the risk budget allocated to asset i. In this case, we can show that the risk budgeting portfolio is the scaled solution of the following optimization problem:

$$x^\star = \arg\min_x \mathcal{R}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i$$

where λ is any positive scalar. Depending on the risk measure, we can use the CCD or the ADMM algorithm.

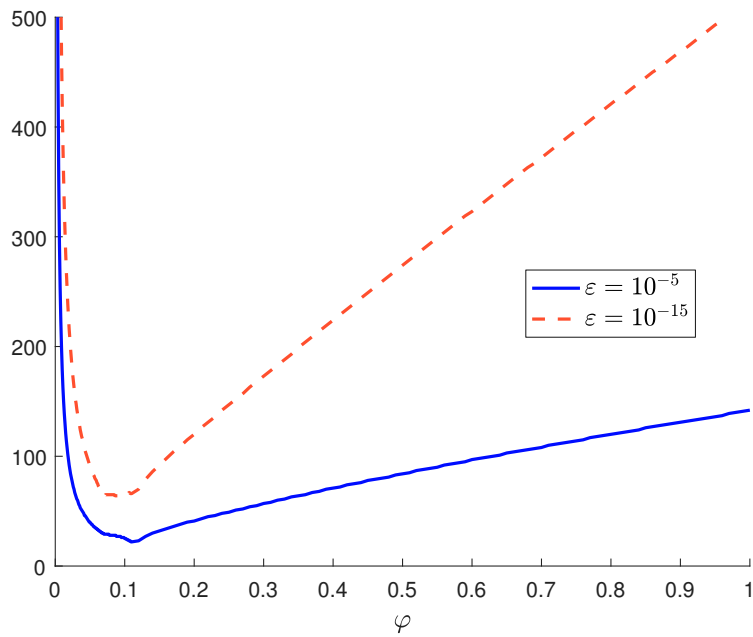


Figure 8.2. *Number of ADMM iterations for finding the ERC portfolio. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

For example, Roncalli (2015) proposes using the standard deviation-based risk measure: $\mathcal{R}(x) = -x^\top(\mu - r) + \xi\sqrt{x^\top \Sigma x}$. In this case, the first-order condition for defining the CCD algorithm is:

$$-(\mu_i - r) + \xi \frac{(\Sigma x)_i}{\sqrt{x^\top \Sigma x}} - \lambda \frac{\mathcal{RB}_i}{x_i} = 0$$

It follows that $\xi x_i (\Sigma x)_i - (\mu_i - r) x_i \sigma(x) - \lambda \sigma(x) \cdot \mathcal{RB}_i = 0$ or equivalently $\alpha_i x_i^2 + \beta_i x_i + \gamma_i = 0$, where $\alpha_i = \xi\sigma_i^2$, $\beta_i = \xi\sigma_i \sum_{j \neq i} x_j \rho_{i,j} \sigma_j - (\mu_i - r)\sigma(x)$ and $\gamma_i = -\lambda\sigma(x) \cdot \mathcal{RB}_i$. We note that the solution x_i depends on the volatility $\sigma(x)$. Here, we face an endogenous problem, because $\sigma(x)$ depends on x_i. Griveau-Billon *et al.* (2015) note that this is not an issue because we may assume that $\sigma(x)$ is almost constant between two coordinate iterations of the CCD algorithm. They deduce that the coordinate solution is then the positive root of the second-degree equation:

$$x_i^{(k+1)} = \frac{-\beta_i^{(k+1)} + \sqrt{\left(\beta_i^{(k+1)}\right)^2 - 4\alpha_i^{(k+1)}\gamma_i^{(k+1)}}}{2\alpha_i^{(k+1)}} \qquad [8.33]$$

where:

$$\begin{cases} \alpha_i^{(k+1)} = \xi\sigma_i^2 \\ \beta_i^{(k+1)} = \xi\sigma_i \left(\sum_{j<i} x_j^{(k+1)} \rho_{i,j}\sigma_j + \sum_{j>i} x_j^{(k)} \rho_{i,j}\sigma_j\right) - (\mu_i - r)\sigma_i^{(k+1)}(x) \\ \gamma_i^{(k+1)} = -\lambda\sigma_i^{(k+1)}(x) \cdot \mathcal{RB}_i \\ \sigma_i^{(k+1)}(x) = \sqrt{\chi^\top \Sigma \chi} \\ \chi = \left(x_1^{(k+1)}, \ldots, x_{i-1}^{(k+1)}, x_i^{(k)}, x_{i+1}^{(k)} \ldots, x_n^{(k)}\right) \end{cases}$$

In the case of the volatility or the standard deviation-based risk measure, we apply the exact formulation of the CCD algorithm because we have an analytical solution of the first-order condition. This is not always the case, especially when we consider skewness-based risk measure (Bruder *et al.* 2016;

Lezmi *et al.* 2018). In this case, we can use the gradient formulation of the CCD algorithm or the ADMM algorithm, which is defined as follows:

$$\begin{cases} x^{(k+1)} = \mathbf{prox}_{\varphi^{-1}\mathcal{R}(x)}\left(y^{(k)} - u^{(k)}\right) \\ v_y^{(k+1)} = x^{(k+1)} + u^{(k)} \\ y^{(k+1)} = \frac{1}{2}\left(v_y^{(k+1)} + \sqrt{v_y^{(k+1)} \odot v_y^{(k+1)} + 4\lambda\varphi^{-1} \cdot \mathcal{RB}}\right) \\ u^{(k+1)} = u^{(k)} + x^{(k+1)} - y^{(k+1)} \end{cases}$$

8.4.2.3. *The most diversified portfolio*

Choueifaty and Coignard (2008) introduce the concept of diversification ratio, which corresponds to the following expression:

$$\mathcal{DR}(x) = \frac{\sum_{i=1}^n x_i\sigma_i}{\sigma(x)} = \frac{x^\top\sigma}{\sqrt{x^\top\Sigma x}}$$

By construction, the diversification ratio of a portfolio fully invested in one asset is equal to 1, whereas it is larger than 1 in the general case. The authors then propose building the most diversified portfolio as the portfolio which maximizes the diversification ratio. It is also the solution to the following minimization problem[24]:

$$\begin{aligned} x^\star \;=\; & \arg\min_x \frac{1}{2}\ln\left(x^\top\Sigma x\right) - \ln\left(x^\top\sigma\right) \\ & \text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ x \in \Omega \end{cases} \end{aligned}$$

This problem is relatively easy to solve using standard numerical algorithms if Ω corresponds to linear constraints, for example, weight constraints. However, the optimal solution may face the same problem as the minimum variance portfolio since most of the time it is concentrated on a small number of assets. Hence, it is interesting to add a weight diversification constraint $\mathcal{D}(x) \geq \mathcal{D}^-$. For example, we can assume that the number of

24 See Choueifaty *et al.* (2013).

effective bets $\mathcal{N}(x)$ is larger than a minimum acceptable value $\mathcal{N}^-$. Contrary to the minimum variance portfolio, we do not obtain a QP problem and we observe that the optimization problem is tricky in practice. Thanks to the ADMM algorithm, we can, however, simplify the optimization problem by splitting the constraints and using the same approach that has been already described on p. 299. The x-update consists of finding the regularized standard MDP:

$$x^{(k+1)} = \arg\min_x \left\{ \frac{1}{2} \ln \left(x^\top \Sigma x \right) - \ln \left(x^\top \sigma \right) + \frac{\varphi}{2} \left\| x - y^{(k)} + u^{(k)} \right\|_2^2 \quad \text{s.t. } \mathbf{1}_n^\top x = 1 \right\}$$

whereas the y-update corresponds to the projection onto the intersection of Ω and $\mathfrak{D} = \{x \in \mathbb{R}^n : \mathcal{D}(x) \geq \mathcal{D}^-\}$:

$$y^{(k+1)} = \mathcal{P}_{\Omega \cap \mathfrak{D}} \left(x^{(k+1)} + u^{(k)} \right)$$

	L/S	**Long-only**					
$\mathcal{N}^-$		0.00	3.00	4.00	5.00	6.00	7.00
$x_1^\star$	41.81	41.04	35.74	30.29	26.08	22.44	18.83
$x_2^\star$	51.88	50.92	43.91	36.68	31.05	26.12	21.19
$x_3^\star$	8.20	8.05	10.12	11.52	12.33	12.80	13.01
$x_4^\star$	−0.43	0.00	2.48	5.12	7.16	8.90	10.51
$x_5^\star$	−0.26	0.00	0.92	2.28	3.60	5.02	6.85
$x_6^\star$	−0.38	0.00	2.03	4.36	6.28	8.02	9.79
$x_7^\star$	−0.51	0.00	3.47	6.68	8.85	10.44	11.65
$x_8^\star$	−0.31	0.00	1.32	3.07	4.65	6.27	8.17
$\mathcal{N}(x)$		2.30	3.00	4.00	5.00	6.00	7.00

Table 8.5. *MDP portfolios (in %)*

We consider the parameter set #2 given in Appendix 8.7.2.2. The results are reported in Table 8.5. The second column corresponds to the long/short MDP portfolio (or $\Omega = \mathbb{R}^n$). By definition, we cannot compute the number of effective bets because it contains short positions. The other columns correspond to the long-only MDP portfolio (or $\Omega = [0, 1]^n$) when we impose

a sufficient number of effective bets $\mathcal{N}^-$. We note that the traditional long-only MDP is poorly diversified in terms of weights since we have $\mathcal{N}(x) = 2.30$. As for the minimum variance portfolio, the MDP tends to the equally weighted portfolio when $\mathcal{N}^-$ tends to the number of assets

8.4.3. *Robo-advisory optimization*

Today's financial industry is facing a digital revolution in all areas: payment services, online banking, asset management, etc. This is particularly true for the financial advisory industry, which has been impacted in the last few years by the emergence of digitalization and robo-advisors. The demand for robo-advisors is strong, which explains the growth of this business[25]. How does one characterize a robo-advisor? This is not simple, but the underlying idea is to build a systematic portfolio allocation in order to provide a customized advisory service. A robo-advisor has two main objectives. The first objective is to know the investor better than a traditional asset manager. Because of this better knowledge, the robo-advisor may propose a more appropriate asset allocation. The second objective is to perform the task in a systematic way and to build an automated rebalancing process. Ultimately, the goal is to offer a customized solution. In fact, the reality is very different. We generally note that many robo-advisors are more a web or a digital application, but not really a robo-advisor. The reason is that portfolio optimization is a very difficult task. In many robo-advisors, asset allocation is then rather human-based or not completely systematic, with the aim to rectify the shortcomings of mean-variance optimization. Over the next five years, the most important challenge for robo-advisors will be to reduce these discretionary decisions and improve the robustness of their systematic asset allocation process. However, this means that robo-advisors must give up the quadratic programming world of the portfolio allocation.

8.4.3.1. *Specification of the objective function*

In order to make mean-variance optimization more robust, two directions can be followed. The first one has been largely explored and adds a penalty function in order to regularize or sparsify the solution (Brodie *et al.* 2009;

25 For instance, the growth was 60% per year in the United States over the last five years. In Europe, the growth is also impressive, even though the market is smaller. In the last two years, assets under management have increased 14-fold.

DeMiguel *et al.* 2009; Carrasco and Noumon 2010; Bruder *et al.* 2013; Bourgeron *et al.* 2018). The second one consists of changing the objective function and considering risk budgeting portfolios instead of mean-variance optimized portfolios (Maillard *et al.* 2010; Roncalli 2013). Even if this second direction has encountered great success, it presents a solution that is not sufficiently flexible in terms of active management. Nevertheless, these two directions are not so divergent. Indeed, Roncalli (2013) shows that the risk budgeting optimization can be viewed as a nonlinear shrinkage approach of the minimum variance optimization. Richard and Roncalli (2015) propose then a unified approach of smart beta portfolios by considering alternative allocation models as penalty functions of the minimum variance optimization. In particular, they use the logarithmic barrier function in order to regularize minimum variance portfolios. This idea has also been reiterated by de Jong (2018) who considers a mean-variance framework.

Therefore, we propose defining the robo-advisor optimization problem as follows:

$$x_{t+1}^{\star} = \arg\min_{x} f_{\mathcal{R}obo}(x) \quad [8.34]$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq \mathbf{1}_n \\ x \in \Omega \end{cases}$$

where:

$$\begin{aligned} f_{\mathcal{R}obo}(x) = {} & \frac{1}{2}(x-b)^\top \Sigma_t (x-b) - \gamma (x-b)^\top \mu_t + \\ & \varrho_1 \left\| \Gamma_1 (x - x_t) \right\|_1 + \frac{1}{2} \varrho_2 \left\| \Gamma_2 (x - x_t) \right\|_2^2 + \\ & \tilde{\varrho}_1 \left\| \tilde{\Gamma}_1 (x - \tilde{x}) \right\|_1 + \frac{1}{2} \tilde{\varrho}_2 \left\| \tilde{\Gamma}_2 (x - \tilde{x}) \right\|_2^2 - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i \end{aligned} \quad [8.35]$$

b is the benchmark portfolio, $\tilde{x}$ is the reference portfolio and x_t is the current portfolio.

This specification is sufficiently broad that it encompasses most models used by the industry. We note that the objective function is made up of three

parts. The first part corresponds to the MVO objective function with a benchmark. If we set b equal to $\mathbf{0}_n$, we obtain the Markowitz utility function. The second part contains $\boldsymbol{\ell}_1$- and $\boldsymbol{\ell}_2$-norm penalty functions. The regularization can be done with respect to the current allocation x_t in order to control the rebalancing process and the smoothness of the dynamic allocation. The regularization can also be done with respect to a reference portfolio, which is generally the strategic asset allocation of the fund. The idea is to control the profile of the fund. For example, if the strategic asset allocation is an 80/20 asset mix policy, we do not want the portfolio to present a defensive or balanced risk profile. Finally, the third part of the objective function corresponds to the logarithmic barrier function, where the parameter λ controls the trade-off between MVO optimization and RB optimization. This last part is very important in order to make the dynamic asset allocation more robust. The hyperparameters of the objective function are ϱ_1, ϱ_2, $\tilde{\varrho}_1$, $\tilde{\varrho}_2$ and λ. They are all positive and can also be set to zero in order to deactivate a penalty function. For instance, ϱ_2 and $\tilde{\varrho}_2$ are equal to zero if we do not want to have a shrinkage of the covariance matrix Σ_t. The hyperparameters ϱ_1 and $\tilde{\varrho}_1$ can also be equal to zero because the $\boldsymbol{\ell}_1$ regularization is generally introduced when specifying the additional constraints Ω. The parameter γ is not really a hyperparameter because it is generally calibrated to target volatility or an expected return. We also note that this model encompasses the Black–Litterman model thanks to the specification of μ_t (Bourgeron *et al.* 2018). Another important component of this framework is the specification of the set $x \in \Omega$. It may include traditional constraints such as weight bounds and/or asset class limits, but we can also add nonlinear constraints such as a turnover limit, an active share floor or a weight diversification constraint.

8.4.3.2. *Derivation of the general algorithm*

Problem [8.34] is equivalent to solving:

$$x_{t+1}^{\star} = \arg\min_{x} f_{\mathcal{R}obo}^{+}(x)$$

where the objective function can be broken down as follows:

$$\begin{aligned} f_{\mathcal{R}obo}^{+}(x) = {} & f_{\text{MVO}}(x) + f_{\boldsymbol{\ell}_1}(x) + f_{\boldsymbol{\ell}_2}(x) + f_{\text{RB}}(x) + \\ & + \Omega_0(x) + \Omega(x) \end{aligned}$$

where:

$$
\begin{aligned}
f_{\text{MVO}}(x) &= \frac{1}{2}(x-b)^{\top}\Sigma_t(x-b) - \gamma(x-b)^{\top}\mu_t \\
f_{\ell_1}(x) &= \varrho_1 \|\Gamma_1(x-x_t)\|_1 + \tilde{\varrho}_1 \left\|\tilde{\Gamma}_1(x-\tilde{x})\right\|_1 \\
f_{\ell_2}(x) &= \frac{1}{2}\varrho_2 \|\Gamma_2(x-x_t)\|_2^2 + \frac{1}{2}\tilde{\varrho}_2 \left\|\tilde{\Gamma}_2(x-\tilde{x})\right\|_2^2 \\
f_{\text{RB}}(x) &= -\lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i
\end{aligned}
$$

and $\Omega_0 = \left\{x \in [0,1]^n : \mathbf{1}_n^{\top} x = 1\right\}$. The ADMM algorithm is implemented as follows:

$$
\begin{aligned}
\{x^{\star}, y^{\star}\} &= \arg\min f_x(x) + f_y(y) \\
&\text{s.t. } x - y = \mathbf{0}_n
\end{aligned}
$$

This is the general approach for solving the robo-advisor problem.

The main task is then to split the function $f^{+}_{\mathcal{R}obo}$ into f_x and f_y. However, in order to be efficient, the x- and y-update steps of the ADMM algorithm must be easy to compute. Therefore, we impose that the x-step is solved using QP or CCD methods, while the y-step is solved using Dykstra's algorithm, where each component corresponds to an analytical proximal operator. Moreover, we also split the set of constraints Ω into a set of linear constraints $\Omega_{\mathcal{L}inear}$ and a set of nonlinear constraints $\Omega_{\mathcal{N}onlinear}$. This leads to the definition of $f_x(x)$ and $f_y(y)$ as follows:

$$
\begin{cases}
f_x(x) = f_{\text{MVO}}(x) + f_{\ell_2}(x) + \quad \Omega_0(x) + \quad \Omega_{\mathcal{L}inear}(x) \\
f_y(y) = f_{\ell_1}(y) + f_{\text{RB}}(x) + \quad \Omega_{\mathcal{N}onlinear}(x)
\end{cases}
\tag{8.36}
$$

We note that $f_x(x)$ has a quadratic form, implying that the x-step may be solved using a QP algorithm. Another formulation is:

$$
\begin{cases}
f_x(x) = f_{\text{MVO}}(x) + f_{\ell_2}(x) + f_{\text{RB}}(x) \\
f_y(y) = f_{\ell_1}(y) + \quad \Omega_0(x) + \quad \Omega_{\mathcal{L}inear}(x) + \quad \Omega_{\mathcal{N}onlinear}(x)
\end{cases}
\tag{8.37}
$$

In this case, the x-step is solved using the CCD algorithm.

8.4.3.3. *Specific algorithms*

8.4.3.3.1. The ADMM-QP formulation

If we consider formulation [8.36], we have:

$$\begin{aligned} f_{\mathrm{QP}}(x) &= f_{\mathrm{MVO}}(x) + f_{\ell_2}(x) \\ &= \frac{1}{2}(x-b)^\top \Sigma_t (x-b) - \gamma (x-b)^\top \mu_t \\ &\quad + \frac{1}{2}\varrho_2 \left\|\Gamma_2 (x - x_t)\right\|_2^2 + \frac{1}{2}\tilde{\varrho}_2 \left\|\tilde{\Gamma}_2 (x - \tilde{x})\right\|_2^2 \\ &= \frac{1}{2}x^\top Q x - x^\top R + C \end{aligned}$$

where $Q = \Sigma_t + \varrho_2 \Gamma_2^\top \Gamma_2 + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2$, $R = \gamma\mu_t + \Sigma_t b + \varrho_2 \Gamma_2^\top \Gamma_2 x_t + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 \tilde{x}$ and C is a constant. Using the fourth ADMM trick, we deduce that $x^{(k+1)}$ is the solution of the following QP problem:

$$\begin{aligned} x^{(k+1)} &= \arg\min_x \frac{1}{2}x^\top (Q + \varphi I_n) x - x^\top \left(R + \varphi\left(y^{(k)} - u^{(k)}\right)\right) \\ &\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq \mathbf{1}_n \end{cases} \end{aligned}$$

Since the proximal operators of f_{ℓ_1} and f_{RB} have already been computed, finding $y^{(k+1)}$ is straightforward with Dykstra's algorithm, as long as the proximal of each nonlinear constraint is known.

8.4.3.3.2. The ADMM-CCD formulation

If we consider formulation [8.37], we have:

$$f_x(x) = f_{\mathrm{QP}}(x) - \lambda \sum_{i=1}^{n} \mathcal{RB}_i \cdot \ln x_i$$

We can show that the CCD algorithm applied to x-update is:

$$x_i^{(k+1)} = \frac{R_i - \sum_{j<i} x_j^{(k+1)} Q_{i,j} - \sum_{j>i} x_j^{(k)} Q_{i,j}}{2Q_{i,i}} + \frac{\sqrt{\left(\sum_{j<i} x_j^{(k+1)} Q_{i,j} + \sum_{j>i} x_j^{(k)} Q_{i,j} - R_i\right)^2 + 4\lambda_i Q_{i,i}}}{2Q_{i,i}}$$

where $Q = \Sigma_t + \varrho_2 \Gamma_2^\top \Gamma_2 + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 + \varphi I_n$, $R = \gamma \mu_t + \Sigma_t b + \varrho_2 \Gamma_2^\top \Gamma_2 x_t + \tilde{\varrho}_2 \tilde{\Gamma}_2^\top \tilde{\Gamma}_2 \tilde{x} + \varphi \left(y^{(k)} - u^{(k)}\right)$ and $\lambda_i = \lambda \cdot \mathcal{RB}_i$. Like the ADMM-QP formulation, the y-update step does not pose any particular difficulties.

8.4.4. *Tips and tricks*

If we consider the different portfolio optimization approaches presented in Table 8.1, we have shown how to solve MVO (1), GMV (2), MDP (3), ERC (4) and RB (5) models. The RQE (7) model is equivalent to the GMV (2) model by replacing the covariance matrix Σ by the dissimilarity matrix D. Below, we implement the Kullback–Leibler model (4) of Bera and Park (2008) using the ADMM framework. Concerning the regularization problems in Table 8.2, ridge (8), lasso (9) and log-barrier (10) penalty functions have already been covered. Indeed, ridge and lasso penalizations correspond to the proximal operator of ℓ_1- and ℓ_2-norm functions by applying the translation $g(x) = x - \tilde{x}$. Shannon's entropy (11) penalization is discussed below. For the constraints that are considered in Table 8.3, imposing no cash and leverage (12) is done with the proximal of the hyperplane $\mathcal{H}_{yperlane}[\mathbf{1}_n, 1]$. No short selling (13) and weight bounds (14) are equivalent to considering the box projections $\mathcal{B}_{ox}[\mathbf{0}_n, \infty]$ and $\mathcal{B}_{ox}[x^-, x^+]$. Asset class limits can be implemented using the projection onto the intersection of two half-spaces $\mathcal{H}_{alfspace}\left[\mathbf{1}_{i \in C_j}, c_j^+\right]$ and $\mathcal{H}_{alfspace}\left[-\mathbf{1}_{i \in C_j}, -c_j^-\right]$. The proximal of the turnover (16) had already been given in equation [8.31]. If we want to impose an upper limit on transaction costs (17), we use the Moreau decomposition and equation [8.30]. Finally, section 8.4.1.1 dealt with the weight diversification problem of the number of active bets. Therefore, it remains to solve leverage limits (18), long/short

exposure (19) restrictions and active management constraints: benchmarking (20), tracking error floor (21) and active share floor (22).

8.4.4.1. *Volatility and return targeting*

We first consider the μ-problem and the σ-problem. Targeting a minimum expected return $\mu(x) \geq \mu^\star$ can be implemented in the ADMM framework using the proximal operator of the hyperplane[26] $\mathcal{H}_{yperlane}[-\mu, -\mu^\star]$. In the case of the σ-problem $\sigma(x) \leq \sigma^\star$, we use the fourth ADMM trick. Let L be the lower Cholesky decomposition of Σ, we have:

$$\sqrt{x^\top \Sigma x} \leq \sigma^\star \Leftrightarrow \sqrt{x^\top (LL^\top) x} \leq \sigma^\star \Leftrightarrow \left\| y^\top y \right\|_2 \leq \sigma^\star$$

where $y = L^\top x$. It follows that the proximal of the y-update is the projection onto the ℓ_2 ball $\mathcal{B}_2(\mathbf{0}_n, \sigma^\star)$.

8.4.4.2. *Leverage management*

If we impose a leverage limit $\sum_{i=1}^n |x_i| \leq \mathcal{L}^+$, we have $\|x\|_1 \leq \mathcal{L}^+$ and the proximal of the y-update is the projection onto the ℓ_1 ball $\mathcal{B}_1(\mathbf{0}_n, \mathcal{L}^+)$. If the leverage constraint concerns the long/short limits $-\mathcal{LS}^- \leq \sum_{i=1}^n x_i \leq \mathcal{LS}^+$, we consider the intersection of the two half-spaces $\mathcal{H}_{alfspace}\left[\mathbf{1}_n, \mathcal{LS}^+\right]$ and $\mathcal{H}_{alfspace}\left[-\mathbf{1}_n, \mathcal{LS}^-\right]$. If we consider an absolute leverage $|\sum_{i=1}^n x_i| \leq \mathcal{L}^+$, we obtain the previous case with $\mathcal{LS}^- = \mathcal{LS}^+ = \mathcal{L}^+$. Portfolio managers can also use another constraint concerning the sum of the k largest values[27]:

$$f(x) = \sum_{i=n-k+1}^{n} x_{(i:n)} = x_{(n:n)} + \ldots + x_{(n-k+1:n)}$$

where $x_{(i:n)}$ is the order statistics of x: $x_{(1:n)} \leq x_{(2:n)} \leq \cdots \leq x_{(n:n)}$. Beck (2017) shows that:

$$\mathbf{prox}_{\lambda f(x)}(v) = v - \lambda \mathcal{P}_\Omega\left(\frac{v}{\lambda}\right)$$

26 We have $\mu(x) \geq \mu^\star \Leftrightarrow x^\top \mu \geq \mu^\star \Leftrightarrow -\mu^\top x \leq -\mu^\star$.

27 An example is the 5/10/40 UCITS rule: A UCITS fund may invest no more than 10% of its net assets in transferable securities or money market instruments issued by the same body, with a further aggregate limitation of 40% of net assets on exposures of greater than 5% to single issuers.

where:

$$\Omega = \left\{ x \in [0,1]^n : \mathbf{1}_n^\top x = k \right\} = \mathcal{B}_{ox}\left[\mathbf{0}_n, \mathbf{1}_n\right] \cap \mathcal{H}_{yperlane}\left[\mathbf{1}_n, k\right]$$

8.4.4.3. *Entropy portfolio and diversification measure*

Bera and Park (2008) propose using a cross-entropy measure as the objective function:

$$x^\star = \arg\min_x \mathrm{KL}\left(x \mid \tilde{x}\right)$$

$$\text{s.t.} \begin{cases} \mathbf{1}_n^\top x = 1 \\ \mathbf{0}_n \leq x \leq \mathbf{1}_n \\ \mu\left(x\right) \geq \mu^\star \\ \sigma\left(x\right) \leq \sigma^\star \end{cases}$$

where $\mathrm{KL}\left(x \mid \tilde{x}\right) = \sum_{i=1}^n x_i \ln\left(x_i / \tilde{x}_i\right)$ and $\tilde{x}$ is a reference portfolio, which is well diversified (e.g. the EW[28] or ERC portfolio). In Appendix 8.7.1.4.2, we show that the proximal operator of $\lambda \mathrm{KL}\left(x \mid \tilde{x}\right)$ is equal to:

$$\mathbf{prox}_{\lambda \mathrm{KL}(v|\tilde{x})}\left(v\right) = \lambda \begin{pmatrix} W\left(\lambda^{-1}\tilde{x}_1 e^{\lambda^{-1}v_1 - \tilde{x}_1^{-1}}\right) \\ \vdots \\ W\left(\lambda^{-1}\tilde{x}_n e^{\lambda^{-1}v_n - \tilde{x}_n^{-1}}\right) \end{pmatrix}$$

where $W\left(x\right)$ is the Lambert W function.

REMARK 8.7.– Using the previous result and the fact that $\mathrm{SE}\left(x\right) = -\mathrm{KL}\left(x \mid \mathbf{1}_n\right)$, we can use Shannon's entropy to define the diversification measure $\mathcal{D}\left(x\right) = \mathrm{SE}\left(x\right)$. Therefore, solving problem [8.28] is straightforward when we consider the following diversification set:

$$\mathfrak{D} = \left\{ x \in [0,1]^n : -\sum_{i=1}^n x_i \ln x_i \geq \mathrm{SE}^- \right\}$$

28 In this case, it is equivalent to maximize Shannon's entropy because $\tilde{x} = \mathbf{1}_n$.

8.4.4.4. *Passive and active management*

In the case of the active share, we use the translation property:

$$\mathcal{AS}(x \mid \tilde{x}) = \frac{1}{2}\sum_{i=1}^{n}|x_i - \tilde{x}_i| = \frac{1}{2}\|x - \tilde{x}\|_1$$

The proximal operator is given in Appendix 8.7.1.5. It is interesting to note that this type of problem cannot be solved using an augmented QP algorithm since it involves the complement of the ℓ_1 ball and not directly the ℓ_1 ball itself. In this case, we face a maximization problem and not a minimization problem and the technique of augmented variables does not work. For tracking error volatility, again we use the fourth ADMM trick:

$$\sigma(x \mid \tilde{x}) = \sqrt{(x - \tilde{x})^\top \Sigma (x - \tilde{x})} = \|y\|_2$$

where $y = L^\top x - L^\top \tilde{x}$. Using our ADMM notations, we have $Ax + By = c$, where $A = L^\top$, $B = -I_n$ and $c = L^\top \tilde{x}$.

8.5. Conclusion

The aim of this chapter is to propose an alternative solution to the quadratic programming algorithm in the context of portfolio allocation. In numerical analysis, the quadratic programming model is a powerful optimization tool, which is computationally very efficient. In portfolio management, the mean-variance optimization model is exactly a quadratic programming model, meaning that it benefits from its computational power. Therefore, the success of the Markowitz allocation model is explained by these two factors: the quadratic utility function and the quadratic programming setup. A lot of academics and professionals have proposed an alternative approach to the MVO framework, but very few of these models are used in practice. The main reason is that these competing models focus on the objective function and not on the numerical implementation. However, we believe that any model which is not tractable will have little success with portfolio managers. The analogy is obvious if we consider the theory of options. The success of the Black–Scholes model lies in the Black–Scholes analytical formula. Over the last 30 years, many models have been created

(e.g. local volatility and stochastic volatility models), but only one can really compete with the Black–Scholes model. This is the SABR model, and the main reason is that it has an analytical formula for implied volatility.

This chapter focuses then on a general approach for numerically solving non-QP portfolio allocation models. For that, we consider some algorithms that have been successfully applied to machine learning and large-scale optimization. For instance, the coordinate descent algorithm is the fastest method for performing high-dimensional lasso regression, while Dykstra's algorithm has been created to find the solution of restricted least squares regression. Since there is a strong link between MVO and linear regression (Scherer 2007), it is not a surprise if these algorithms can help solve regularized MVO allocation models. However, these two algorithms are not sufficient for defining a general framework. For that, we need to use the alternating direction method of multipliers and proximal gradient methods. Finally, the combination of these four algorithms (CD, ADMM, PO and Dykstra) allows us to consider allocation models that cannot be cast into a QP form.

In this chapter, we have first considered allocation models with non-quadratic objective functions. For example, we have used models based on the diversification ratio, Shannon's entropy or the Kullback–Leibler divergence. Second, we have solved regularized MVO models with nonlinear penalty functions such as the ℓ_p-norm penalty or the logarithmic barrier. Third, we have discussed how to handle nonlinear constraints. For instance, we have imposed constraints on active share, volatility targeting, leverage limits, transaction costs, etc. Most importantly, these three non-QP extensions can be combined.

With the development of quantitative strategies (smart beta, factor investing, alternative risk premia, systematic strategies, robo-advisors, etc.), the asset management industry has dramatically changed over the last five years. This is just the beginning and we think that alternative data, machine learning methods and artificial intelligence will massively shape investment processes in the future. This chapter is an illustration of this trend and shows how machine learning optimization algorithms allow us to move away from the traditional QP world of portfolio management.

8.6. Acknowledgements

We would like to thank Mohammed El Mendili, Edmond Lezmi, Lina Mezghani, Jean-Charles Richard, Jules Roche and Jiali Xu for their helpful comments.

8.7. Appendix

8.7.1. *Mathematical results*

8.7.1.1. *Augmented QP formulation of the turnover management problem*

The augmented QP problem is defined by:

$$X^\star = \arg\min_X \frac{1}{2} X^\top Q X - X^\top R$$

$$\text{s.t.} \begin{cases} AX = B \\ CX \leq D \\ \mathbf{0}_{3n} \leq X \leq \mathbf{1}_{3n} \end{cases}$$

where $X = \left(x_1, \ldots, x_n, x_1^-, \ldots, x_n^-, x_1^+, \ldots, x_n^+\right)$ is a $3n \times 1$ vector, Q is a $3n \times 3n$ matrix:

$$Q = \begin{pmatrix} \Sigma & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \end{pmatrix}$$

$R = (\gamma\mu, \mathbf{0}_n, \mathbf{0}_n)$ is a $3n \times 1$ vector, A is a $(n+1) \times 3n$ matrix:

$$A = \begin{pmatrix} \mathbf{1}_n^\top & \mathbf{0}_n^\top & \mathbf{0}_n^\top \\ I_n & I_n & -I_n \end{pmatrix}$$

$B = (1, \bar{x})$ is a $(n+1) \times 1$ vector, $C = \begin{pmatrix} \mathbf{0}_n^\top & \mathbf{1}_n^\top & \mathbf{1}_n^\top \end{pmatrix}$ is a $1 \times 3n$ matrix and $D = \boldsymbol{\tau}^+$.

8.7.1.2. *QP problem with a hyperplane constraint*

We consider the following QP problem:

$$x^\star = \arg\min_x \frac{1}{2} x^\top Q x - x^\top R \qquad \text{s.t.} \quad a^\top x = b$$

The associated Lagrange function is $\mathcal{L}(x;\lambda) = \frac{1}{2} x^\top Q x - x^\top R + \lambda \left(a^\top x - b\right)$. The first-order conditions are $\partial_x \mathcal{L}(x;\lambda) = Qx - R + \lambda a = \mathbf{0}_n$ and $\partial_\lambda \mathcal{L}(x;\lambda) = a^\top x - b = 0$. We obtain $x = Q^{-1}(R + \lambda a)$. Because $a^\top x - b = 0$, we have $a^\top Q^{-1} R + \lambda a^\top Q^{-1} a = b$ and:

$$\lambda^\star = \frac{b - a^\top Q^{-1} R}{a^\top Q^{-1} a}$$

The optimal solution is then:

$$x^\star = Q^{-1}\left(R + \frac{b - a^\top Q^{-1} R}{a^\top Q^{-1} a} a\right)$$

8.7.1.3. *Derivation of the soft-thresholding operator*

We consider the equation $cx - v + \lambda \partial |x| \in 0$, where $c > 0$ and $\lambda > 0$. Since we have $\partial |x| = \operatorname{sign}(x)$, we deduce that:

$$x^\star = \begin{cases} c^{-1}(v + \lambda) & \text{if } x^\star < 0 \\ 0 & \text{if } x^\star = 0 \\ c^{-1}(v - \lambda) & \text{if } x^\star > 0 \end{cases}$$

If $x^\star < 0$ or $x^\star > 0$, then we have $v + \lambda < 0$ or $v - \lambda > 0$. This is equivalent to set $|v| > \lambda > 0$. The case $x^\star = 0$ implies that $|v| \leq \lambda$. We deduce that $x^\star = c^{-1} \cdot \mathcal{S}(v;\lambda)$, where $\mathcal{S}(v;\lambda)$ is the soft-thresholding operator:

$$\begin{aligned} \mathcal{S}(v;\lambda) &= \begin{cases} 0 & \text{if } |v| \leq \lambda \\ v - \lambda \operatorname{sign}(v) & \text{otherwise} \end{cases} \\ &= \operatorname{sign}(v) \cdot (|v| - \lambda)_+ \end{aligned}$$

8.7.1.4. *Proximal operators*

8.7.1.4.1. Projection onto the intersection of a ℓ_2 ball and a box

We note $f_1(x) = \quad_{\Omega_1}(x)$ and $f_2(x) = \quad_{\Omega_2}(x)$, where $\Omega_1 = \mathcal{B}_2(c, r)$ and $\Omega_2 = \mathcal{B}_{ox}[x^-, x^+] = \{x \in \mathbb{R}^n : x^- \leq x \leq x^+\}$. Dykstra"s algorithm becomes:

$$\begin{cases} x^{(k+1)} = c + \dfrac{r}{\max\left(r, \left\|y^{(k)} + z_1^{(k)} - c\right\|_2\right)}\left(y^{(k)} + z_1^{(k)} - c\right) \\ z_1^{(k+1)} = y^{(k)} + z_1^{(k)} - x^{(k+1)} \\ y^{(k+1)} = \mathcal{T}\left(x^{(k+1)} + z_2^{(k)}; x^-, x^+\right) \\ z_2^{(k+1)} = x^{(k+1)} + z_2^{(k)} - y^{(k+1)} \end{cases}$$

This algorithm is denoted by $\mathcal{P}_{\mathcal{B}\text{ox}-\mathcal{B}\text{all}}(v; x^-, x^+, c, r)$.

8.7.1.4.2. Shannon's entropy and Kullback–Leibler divergence

If we consider the scalar function $f(x) = \lambda x \ln(x/\tilde{x})$, where $\tilde{x}$ is a constant, we have:

$$\lambda f(x) + \frac{1}{2}\|x - v\|_2^2 = \lambda x \ln\frac{x}{\tilde{x}} + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

The first-order condition is:

$$\lambda\frac{1}{\tilde{x}} + \lambda \ln\frac{x}{\tilde{x}} + x - v = 0 \Leftrightarrow \left(\lambda^{-1}x\right)e^{\left(\lambda^{-1}x\right)} = \lambda^{-1}\tilde{x}e^{\lambda^{-1}v - \frac{1}{\tilde{x}}}$$

We deduce that the root is equal to $x^\star = \lambda W\left(\frac{\tilde{x}e^{\lambda^{-1}v - \frac{1}{\tilde{x}}}}{\lambda}\right)$, where $W(x)$ is the Lambert W function satisfying $W(x)e^{W(x)} = x$. In the case of the Kullback–Leibler divergence $\mathrm{KL}(x) = \sum_{i=1}^n x_i \ln(x_i/\tilde{x}_i)$, it follows that:

$$\mathbf{prox}_{\lambda\,\mathrm{KL}(v|\tilde{x})}(v) = \lambda\begin{pmatrix} W\left(\lambda^{-1}\tilde{x}_1 e^{\lambda^{-1}v_1 - \tilde{x}_1^{-1}}\right) \\ \vdots \\ W\left(\lambda^{-1}\tilde{x}_n e^{\lambda^{-1}v_n - \tilde{x}_n^{-1}}\right) \end{pmatrix}$$

The proximal of Shannon's entropy $\mathrm{SE}(x) = -\sum_{i=1}^{n} x_i \ln x_i$ is a special case of the previous result[29] with $\tilde{x}_i = 1$.

8.7.1.4.3. Projection onto the complement $\bar{\mathcal{B}}_2(c, r)$ of the ℓ_2 ball

We consider the proximal problem where $\Omega = \{x \in \mathbb{R}^n : \|x - c\|_2 \geq r\}$:

$$\begin{aligned} x^\star &= \arg\min_x \frac{1}{2}(x-v)^\top (x-v) \\ &\text{s.t. } (x-c)^\top (x-c) - r^2 \geq 0 \end{aligned}$$

We deduce that the Lagrange function is equal to:

$$\mathcal{L}(x;\lambda) = \frac{1}{2}(x-v)^\top (x-v) - \lambda\left((x-c)^\top (x-c) - r^2\right)$$

The first-order condition is $\partial_x \mathcal{L}(x;\lambda) = x - v - 2\lambda(x-c) = \mathbf{0}_n$, whereas the KKT condition is $\min\left(\lambda, (x-c)^\top (x-c) - r^2\right) = 0$. We distinguish two cases:

1) if $\lambda = 0$, this means that $x^\star = v$ and $(x-c)^\top (x-c) - r^2 > 0$;

2) if $\lambda > 0$, we have $(x-c)^\top (x-c) = r^2$. Then, we obtain the following system:

$$\begin{cases} x - v - 2\lambda(x-c) = \mathbf{0}_n \\ (x-c)^\top (x-c) = r^2 \end{cases}$$

We deduce that:

$$(x-c) - (v-c) - 2\lambda(x-c) = \mathbf{0}_n \qquad [8.38]$$

and:

$$\lambda^\star = \frac{r^2 - (x-c)^\top (v-c)}{2r^2}$$

We note that:

$$[8.38] \Leftrightarrow (x-c)^\top (v-c)(x-c) = r^2 (v-c)$$

29 We use the fact that $\max \mathrm{SE}(x) = \min -\mathrm{SE}(x)$.

Because $(x-c)^\top (v-c)$ is a scalar, we deduce that $x-c$ and $v-c$ are two collinear vectors. The optimal solution is:

$$x^\star = c + r\frac{(v-c)}{\|v-c\|_2}$$

Combining the two cases gives:

$$\mathbf{prox}_{1_{\Omega(x)}}(v) = c + \frac{r}{\min(r, \|v-c\|_2)}(v-c)$$

This is the formula of the projection onto the ℓ_2 ball, but the minimum function has replaced the maximum function.

8.7.1.5. *Projection onto the complement $\bar{\mathcal{B}}_1(c,r)$ of the ℓ_1 ball*

This proximal problem associated with $\bar{\mathcal{B}}_1(\mathbf{0}_n, r)$ is:

$$x^\star = \arg\min_x \frac{1}{2}(x-v)^\top (x-v) \qquad \text{s.t.} \quad \|x\|_1 \geq r$$

We deduce that the Lagrange function is equal to:

$$\mathcal{L}(x;\lambda) = \frac{1}{2}(x-v)^\top (x-v) - \lambda(\|x\|_1 - r)$$

The first-order condition is $\partial_x \mathcal{L}(x;\lambda) = x - v - \lambda \operatorname{sign}(x) = \mathbf{0}_n$, whereas the KKT condition is $\min(\lambda, \|x\|_1 - r) = 0$. We distinguish two cases. If $\lambda = 0$, this means that $x^\star = v$ and $\|x\|_1 \geq r$. If $\lambda > 0$, we have $\|x\|_1 = r$. Then, we obtain the following system of equations: $x - v = \lambda \operatorname{sign}(x)$ and $\|x\|_1 = r$. The first condition gives that $x - v$ is a vector whose elements are $+\lambda$ and/or $-\lambda$, whereas the second condition shows that x is on the surface of the ℓ_1 ball. Unfortunately, there is no unique solution. Hence, we assume that $\operatorname{sign}(x) = \operatorname{sign}(v)$ and we modify the sign function: $\operatorname{sign}(a) = 1$ if $a \geq 0$ and $\operatorname{sign}(a) = -1$ if $a < 0$. In this case, there is a unique solution $x^\star = v + \lambda^\star \operatorname{sign}(v)$, where $\lambda^\star = n^{-1}(r - \|v\|_1)$ because $|v + \lambda \operatorname{sign}(v)| = |v| + \lambda$. Combining the two cases implies that:

$$\mathcal{P}_{\bar{\mathcal{B}}_1(\mathbf{0}_n,r)}(v) = \begin{cases} v & \text{if } \|v\|_1 \geq r \\ v + \operatorname{sign}(v) \odot n^{-1}(r - \|v\|_1) & \text{if } \|v\|_1 < r \end{cases}$$

Using the translation property, we deduce that:

$$\mathcal{P}_{\bar{\mathcal{B}}_1(c,r)}(v) = v + \operatorname{sign}(v-c) \odot \frac{\max\left(r - \|v-c\|_1, 0\right)}{n}$$

8.7.1.6. *The bid-ask linear cost function*

If we consider the scalar function $f(x) = \alpha(\gamma - x)_+ + \beta(x-\gamma)_+$, we have:

$$f_v(x) = \lambda\alpha(\gamma - x)_+ + \lambda\beta(x-\gamma)_+ + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$$

Following Beck (2017), we distinguish three cases:

1) If $f_v(x) = \lambda\alpha(\gamma - x) + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$, then $f_v'(x) = -\lambda\alpha + x - v$ and $x^\star = v + \lambda\alpha$. This implies that $\gamma - x^\star > 0$ or $v < \gamma - \lambda\alpha$.

2) If $f_v(x) = \lambda\beta(x-\gamma)_+ + \frac{1}{2}x^2 - xv + \frac{1}{2}v^2$, then $f_v'(x) = \lambda\beta + x - v$ and $x^\star = v - \lambda\beta$. This implies that $x^\star - \gamma > 0$ or $v < \gamma + \lambda\beta$.

3) If $v \in [\gamma - \lambda\alpha, \gamma + \lambda\beta]$, the minimum is not obtained at a point of differentiability. Since γ is the only point of non-differentiability, we obtain $x^\star = \gamma$.

Therefore, we can write the proximal operator in the following compact form:

$$x^\star = \gamma + (v - \gamma - \lambda\beta)_+ - (v - \gamma + \lambda\alpha)_-$$

where x_- and x_+ are the negative part and the positive part of x, respectively. If we consider the vector-value function $f(x) = \sum_{i=1}^n \alpha_i(\gamma_i - x_i)_+ + \beta_i(x_i - \gamma_i)_+$, we deduce that:

$$\mathbf{prox}_{\lambda f(x)}(v) = \gamma + \mathcal{S}(v - \gamma; \lambda\alpha, \lambda\beta)$$

where $\mathcal{S}(v; \lambda_-, \lambda_+) = (v - \lambda_+)_+ - (v + \lambda_-)_-$ is the two-sided soft-thresholding operator.

8.7.2. *Data*

8.7.2.1. *Parameter set #1*

We consider a capitalization-weighted stock index, which is composed of eight stocks. The weights of this benchmark are equal to 23%, 19%, 17%, 9%, 8%, 6% and 5%. We assume that their volatilities are 21%, 20%, 40%, 18%, 35%, 23%, 7% and 29%. The correlation matrix is defined as follows:

$$\rho = \begin{pmatrix} 100\% & & & & & & & \\ 80\% & 100\% & & & & & & \\ 70\% & 75\% & 100\% & & & & & \\ 60\% & 65\% & 90\% & 100\% & & & & \\ 70\% & 50\% & 70\% & 85\% & 100\% & & & \\ 50\% & 60\% & 70\% & 80\% & 60\% & 100\% & & \\ 70\% & 50\% & 70\% & 75\% & 80\% & 50\% & 100\% & \\ 60\% & 65\% & 70\% & 75\% & 65\% & 70\% & 80\% & 100\% \end{pmatrix}$$

8.7.2.2. *Parameter set #2*

We consider a universe of eight stocks. We assume that their volatilities are 25%, 20%, 15%, 18%, 30%, 20%, 15% and 35%. The correlation matrix is defined as follows:

$$\rho = \begin{pmatrix} 100\% & & & & & & & \\ 20\% & 100\% & & & & & & \\ 55\% & 60\% & 100\% & & & & & \\ 60\% & 60\% & 60\% & 100\% & & & & \\ 60\% & 60\% & 60\% & 60\% & 100\% & & & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 100\% & & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 100\% & \\ 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 60\% & 100\% \end{pmatrix}$$

8.8. References

Bauschke, H.H. and Borwein, J.M. (1994). Dykstra's alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3), 418–443.

Beale, E.M.L. (1959). On quadratic programming. *Naval Research Logistics Quarterly*, 6(3), 227–243.

Beck, A. (2017). *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization, 25, SIAM.

Bera, A.K. and Park, S.Y. (2008). Optimal portfolio diversification using the maximum entropy principle. *Econometric Reviews*, 27(4–6), 484–512.

Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Bourgeron, T., Lezmi, E., and Roncalli, T. (2018). Robust asset allocation for Robo-Advisors. *arXiv*, 1902.05710.

Brodie, J., Daubechies, I., De Mol, C., Giannone, D., and Loris, I. (2009). Sparse and stable Markowitz portfolios. *Proceedings of the National Academy of Sciences*, 106(30), 12267–12272.

Bruder, B., Gaussel, N., Richard, J-C., and Roncalli, T. (2013). Regularization of portfolio allocation. *SSRN*. Available at: www.ssrn.com/abstract=2767358.

Bruder, B., Kostyuchyk, N., and Roncalli, T. (2016). Risk parity portfolios with skewness risk: An application to factor investing and alternative risk Premia. *SSRN*. Available at: www.ssrn.com/abstract=2813384.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2010). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1), 1–122.

Carmichael, B., Koumou, G.B., and Moran, K. (2018). Rao's quadratic entropy and maximum diversification indexation. *Quantitative Finance*, 18(6), 1017–1031.

Carrasco, M., and Noumon, N. (2010). Optimal portfolio selection using regularization, University of Montreal. Discussion Paper.

Choueifaty, Y. and Coignard, Y. (2008). Toward maximum diversification. *Journal of Portfolio Management*, 35(1), 40–51.

Choueifaty, Y., Froidure, T. and Reynier, J. (2013). Properties of the most diversified portfolio. *Journal of Investment Strategies*, 2(2), 49–70.

Combettes, P.L. and Pesquet, J.C. (2011). Proximal splitting methods in signal processing. In *Fixed-point Algorithms for Inverse Problems in Science and Engineering*, Bauschke, H.H., Burachik, R.S., Combettes, P.L., Elser, V., Luke, D.R., and Wolkowicz, H. (eds). Springer Optimization and Its Applications, 48, Springer, New York.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

Cottle, R.W. and Infanger, G. (2010). Harry Markowitz and the early history of quadratic programming. In *Handbook of Portfolio Construction*, Guerard, J.B. (ed.). Springer, New York.

de Jong, M. (2018). Portfolio optimisation in an uncertain world. *Journal of Asset Management*, 19(4), 216–221.

DeMiguel, V., Garlappi, L., Nogales, F.J., and Uppal, R. (2009). A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. *Management Science*, 55(5), 798–812.

Dykstra, R.L. (1983). An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384), 837–842.

Frank, M., and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3, 95–110.

Gabay, D., and Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1), 17–40.

Gonzalvez, J., Lezmi, E., Roncalli, T., and Xu, J. (2019). Financial applications of Gaussian processes and bayesian optimization. *arXiv*. Available at: arxiv.org/abs/1903.04841.

Gould, N.I.M. and Toint, P.L. (2000). A quadratic programming bibliography. *Numerical Analysis Group Internal Report*, 1, 142.

Griveau-Billion, T., Richard, J-C., and Roncalli, T. (2013). A fast algorithm for computing high-dimensional risk parity portfolios. *SSRN*. Available at: www.ssrn.com/abstract=2325255.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning*. 2nd edition, Springer, New York.

Jacobs, R.A. (1988). Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1(4), 295–307.

Jagannathan, R. and Ma, T. (2003). Risk reduction in large portfolios: Why imposing the wrong constraints helps. *Journal of Finance*, 58(4), 1651–1684.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4), 541–551.

Lezmi, E., Malongo, H., Roncalli, T., and Sobotka, R. (2018). Portfolio allocation with skewness risk: A practical guide. *SSRN*. Available at: www.ssrn.com/abstract=3201319.

Maillard, S., Roncalli, T. and Teïletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, 36(4), 60–70.

Mann, H.B. (1943). Quadratic forms with linear constraints. *American Mathematical Monthly*, 50, 430–433.

Markowitz, H. (1952). Portfolio selection. *Journal of Finance*, 7(1), 77–91.

Markowitz, H. (1956). The optimization of a quadratic function subject to linear constraints. *Naval Research Logistics Quarterly*, 3(1–2), 111–133.

Michaud, R.O. (1989). The Markowitz optimization enigma: Is 'optimized' optimal? *Financial Analysts Journal*, 45(1), 31–42.

Nesterov, Y.E. (1983). A method for solving the convex programming problem with convergence rate $O\left(k^{-2}\right)$. *Doklady Akademii Nauk SSSR*, 269, 543–547.

Nesterov, Y.E. (2012). Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2), 341–362.

Parikh, N., and Boyd, S. (2014). Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3), 127–239.

Polyak, B.T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5), 1–17.

Qian, E. (2005). Risk parity portfolios: Efficient portfolios through true diversification. *Panagora Asset Management*, September.

Richard, J-C. and Roncalli, T. (2015). Smart beta: Managing diversification of minimum variance portfolios. In *Risk-based and Factor Investing*, Jurczenko, E. (ed.), ISTE Press, London and Elsevier Oxford.

Richard, J-C. and Roncalli, T. (2019). Constrained risk budgeting portfolios: Theory, algorithms, applications & puzzles. *arXiv*, 1902.05710.

Roncalli, T. (2013). *Introduction to Risk Parity and Budgeting*, Chapman & Hall/CRC Financial Mathematics Series, Boca Raton.

Roncalli, T. (2015). Introducing expected returns into risk parity portfolios: A new framework for asset allocation. *Bankers, Markets & Investors*, 138, 18–28.

Scherer, B. (2007). *Portfolio Construction & Risk Budgeting*. 3rd edition. Risk Books, London.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society B*, 58(1), 267–288.

Tibshirani, R.J. (2017). Dykstra's algorithm, ADMM, and coordinate descent: Connections, insights, and extensions. In *Advances in Neural Information Processing Systems*, Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. and Garnett, R. (eds), MIT Press, Massachusetts.

Tseng, P. (2001). Convergence of a block coordinate descent method for Nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3), 475–494.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley & Sons, New York.

Wolfe, P. (1959). The simplex method for quadratic programming. *Econometrica*, 27(3), 382–398.

Wright, S.J. (2015). Coordinate descent algorithms. *Mathematical Programming*, 151(1), 3–34.

Yu, J.R., Lee, W.Y., and Chiou, W.J.P. (2014). Diversified portfolios with different entropy measures. *Applied Mathematics and Computation*, 241, 47–63.

9

Hierarchical Risk Parity: Accounting for Tail Dependencies in Multi-asset Multi-factor Allocations

We investigate portfolio diversification strategies based on hierarchical clustering. These hierarchical risk parity strategies use graph theory and unsupervised machine learning to build diversified portfolios by acknowledging the hierarchical structure of the investment universe. In this chapter, we consider two dissimilarity measures for clustering a multi-asset multi-factor universe. While the Pearson correlation coefficient is a popular choice, we are especially interested in a measure based on the lower tail dependence coefficient. Such innovation is expected to achieve better tail risk management in the context of allocating to skewed style factor strategies. Indeed, the corresponding hierarchical risk parity strategies seem to have been navigating the associated downside risk better, yet come at the cost of high turnover. A comparison based on block-bootstrapping evidences alternative risk parity strategies along economic factors to be on par, in terms of downside risk, with those based on statistical clusters.

In an attempt to construct more efficient and better risk-managed portfolios, investors have recently turned to diversifying their portfolios through factors rather than traditional asset classes. We investigate meaningful ways of generating a coherent multi-asset multi-factor allocation that is able to harvest the associated asset and factor premia in a balanced fashion. Standard portfolio theory would suggest aiming for an optimal risk and return trade-off by resorting to the seminal mean-variance paradigm of Markowitz (1952). Yet, given the notorious sensitivity of such portfolio

Chapter written by Harald LOHRE, Carsten ROTHER and Kilian Axel SCHÄFER.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

optimization with regard to expected return inputs, we may rather disregard forecasting returns and focus on estimating risk. As a result, researchers have developed various risk-based allocation strategies in pursuit of portfolio diversification.

An innovative risk-based strategy for managing diversification was introduced by Meucci (2009). Conducting a principal component analysis (PCA), he aims to identify the main risk drivers in a given set of assets. The orthogonal eigenvectors resulting from the PCA can be viewed as principal portfolios representing uncorrelated risk sources, and a portfolio is considered well-diversified if the overall risk is distributed equally across these uncorrelated principal portfolios. Given the statistical nature of PCA, Meucci *et al*. (2015) suggest resorting to a minimum-torsion transformation instead. The resulting uncorrelated risk sources are economically more meaningful. Along these lines, the literature has advanced diversified risk parity strategies designed to maximize diversification benefits across asset classes and style factors, see Lohre *et al*. (2014) and Bernardi *et al*. (2018).

While such an approach and its outcome are dependent on choosing and designing an appropriate risk model, the recent literature has presented risk parity allocation paradigms guided by hierarchical clustering techniques – prompting Lopez de Prado (2016) to label the technique hierarchical risk parity (HRP). Given a set of asset class and style factor returns, the corresponding algorithm would cluster these according to some distance metric and then allocate equal risk budgets along these clusters. Such clusters might be deemed more natural building blocks than the aggregate risk factors in that they automatically pick up the dependence structure and form meaningful ingredients to aid portfolio diversification.

The contribution of this chapter is to thoroughly examine the use and merits of hierarchical clustering techniques in the context of multi-asset multi-factor investing. In particular, it will contrast these techniques with several competing risk-based allocation paradigms, such as $1/N$, minimum-variance, standard risk parity and diversified risk parity. A major innovation is investigating HRP strategies based on tail dependence clustering, as opposed to standard correlation-based clustering. Such an approach might be particularly relevant given the elevated tail risk of some style factors. Hierarchical risk parity strategies generally build on two steps:

first, hierarchical clustering algorithms uncover a hierarchical structure of the considered investment universe, resulting in a tree-based representation. Second, the portfolio weights result from applying an allocation strategy along the hierarchical structure, and thus the overall portfolio is expected to exhibit a meaningful degree of diversification.

As estimates of covariance or correlation matrices are subject to estimation errors, correlation-based clusters and networks are meaningful for constructing diversified portfolios, because these can have a filtering effect, resulting in more reliable outcomes, see Tumminello *et al*. (2010). In this vein, Lopez de Prado (2016) argues that the correlation matrix is too complex to be fully analyzed and lacks a hierarchical structure. Instead of analyzing the full covariance matrix, he suggests considering the corresponding minimum spanning tree (MST) with N nodes (one node for each asset) and only $N - 1$ edges, i.e. focusing on the most relevant correlations. Deriving the MST requires the definition of a distance measure, which is often referred to as a dissimilarity measure. The MST is naturally linked to the hierarchical clustering algorithm, which is called single linkage. In a direct way, the MST gives the hierarchical organization of the investigated assets and the optimal portfolio weights can be derived by applying an allocation scheme to the hierarchical structure.

Given a multi-asset multi-factor investment universe, we must keep in mind that the return distributions of the underlying factor strategies can be highly skewed. A well-known investment style is the FX carry trade, which stipulates buying currencies with the highest short-term interest rates and selling those with the lowest short-term interest rates. While the return pattern of the FX carry trade is benign in calm markets, it happens to be prone to sudden drawdowns in stress episodes, when FX investors shy away from more risky currencies. Obviously, this is a prime example as to why tail risk is an important consideration in the context of constructing style factor allocations. Therefore, this chapter considers portfolio construction techniques that specifically incorporate the notion of tail risk management. While the above risk parity allocation paradigm is ultimately centered around maintaining equal-volatility risk budgets, we may wonder as to whether alternative approaches might be more appropriate to navigate the non-normality of factor returns. In that regard, Boudt *et al*. (2013) put

forward equalizing risk contributions to the portfolio's CVaR[1]. This approach thus avoids risk concentrations in the portfolio's tail risk and could simply be thought of as tail risk parity. Yet, the underlying portfolio optimization is more cumbersome and less robust than the traditional risk parity algorithm. In that sense, we will seek to apply the robust hierarchical clustering based on tail dependency measures above to explicitly account for tail risk while not sacrificing the robustness of portfolio weights.

A variety of other allocation strategies based on hierarchical clustering were developed using different linkage methods, allocation schemes and (dis-)similarity measures; see, for instance, Raffinot (2017). These measures usually build on the correlation coefficient. To acknowledge tail risk, other scholars proposed measures focusing on the association among assets during times of large losses. One possible choice is the lower tail dependence coefficient; see, for instance, De Luca and Zuccolotto (2011) and Durante *et al*. (2015). Building upon the hierarchical risk parity of Lopez de Prado (2016), we propose and investigate innovative variations of this strategy using the lower tail dependence coefficient.

The chapter is organized as follows: section 9.1 illustrates hierarchical clustering in the context of the multi-asset multi-factor universe. We also introduce hierarchical risk parity strategies based on the Pearson correlation coefficient. Section 9.2 introduces hierarchical clustering based on the lower tail dependence coefficient. Section 9.3 provides an overview of traditional risk-based allocation strategies and outlines a framework to measure and manage portfolio diversification. Section 9.4 examines the performance of the introduced HRP strategies relative to the traditional alternatives. Section 9.5 concludes.

9.1. Hierarchical risk parity strategies

Investigating hierarchical risk parity strategies in the context of multi-asset multi-factor investing, the respective market and style factor universe is discussed. Afterwards, we illustrate the concept of hierarchical risk parity by conveying the inherent hierarchical structure.

1 Recently, Jurczenko and Teiletche (2019) have proposed an expected shortfall risk budgeting framework that accounts for risks beyond volatility, including asymmetry and tail risk.

9.1.1. *The multi-asset multi-factor universe*

We consider multi-asset multi-factor data that combines the traditional asset classes equity, bond, commodity and credit as well as different style factors. The times series are available for the period from March 3, 2003 to December 29, 2017. The global equity and bond markets are represented by equity index futures for S&P 500, Nikkei 225, FTSE 100, EuroSTOXX 50 and MSCI Emerging Markets, and bond index futures for US 10Y Treasuries, Bund, Japanese Government Bonds (JGB) 10Y and UK gilts. The credit risk premium is captured by the Bloomberg Barclays US Corporate Investment Grade (Credit IG) and High Yield (Credit HY) indices, both of which are duration-hedged to synthesize pure credit risk. Gold, oil, copper and agriculture indices are chosen to cover the commodity market.

Style factors systematically follow specific investment styles. In total, we consider the five investment styles carry, value, momentum, quality and volatility carry within the asset classes equity, (interest) rates, commodity and foreign exchange. Carry is based on the idea that high-yield assets tend to outperform low-yield assets, while value investing is based on the idea that cheap assets (according to a given valuation metric) tend to outperform expensive assets momentum investors assume that recent winning assets outperform recent losing assets. Quality (or defensive) investing builds on the observation that high-quality assets tend to have higher risk-adjusted returns than low-quality assets. Finally, the volatility carry style is based on the assumption that long-volatility assets will decline in value due to an upward sloping forward volatility curve. We source the underlying return time series from Goldman Sachs (GS) and Invesco Quantitative Strategies (IQS). The exact factor definitions as provided by GS and IQS can be found in section 9.7.

For measuring diversification, an appropriate factor model along suitable economic factors can be established. To benchmark the statistical clusters vis-à-vis economic factors, we will include the market factors equity, duration and commodity. Further, taking a pure style factor investing perspective, we build aggregate style factors across asset classes, i.e. the aggregate momentum style factor would be based on equity momentum, FX momentum, rates momentum and commodity momentum. In the same vein, we construct aggregate carry, value, quality and volatility carry factors. The aggregated market and style

factors each apply a weighting scheme that ensures equal risk contributions of the underlying asset classes or style factors.

9.1.2. *The hierarchical multi-asset multi-factor structure*

Many portfolio optimization methods like the Markowitz mean-variance approach are sensitive to changes in input variables, and small estimation errors can lead to vast differences in optimal portfolio allocations. Lopez de Prado (2016) asserts that correlation and covariance matrices are simply too complex to be fully analyzed, and disregard the hierarchical structure of interactions among assets. The full variance-covariance matrix can be visualized by a complete graph with N nodes and $\frac{1}{2}N(N-1)$ edges; see Figure 9.1 (left). To reduce complexity, we would want to focus on relevant correlations by dropping unnecessary edges. In this regard, a well-known approach from graph theory is the so-called minimum spanning tree (MST). The MST is a subset of $N-1$ edges of the original graph, connecting all vertices without any cycles and with the minimum total edge weight. An algorithm for obtaining the MST was introduced by Prim (1957). Before applying this algorithm, we have to define a distance measure, which is often based on the correlation coefficient. We will refer to this measure as the dissimilarity measure, since it aims to measure the dissimilarity of the assets (and factors). A precise definition is provided in section 9.1.3. Applying the dissimilarity measure to the correlation matrix leads to the so-called dissimilarity matrix and allows us to derive the MST; see Figure 9.1 (right). As this is based on correlations, it is often referred to as a correlation network.

We successfully reduced the complete graph with $\frac{1}{2}N(N-1)$ edges to a connected graph with $N-1$ edges, which focuses on essential information contained in the correlation matrix and introduces a hierarchical structure. Looking at the branches, we identify a commodity cluster (top right), an equity style cluster (top left) and a volatility style cluster (bottom left).

Graph theory is linked to unsupervised machine learning. In particular, the MST is naturally related to the hierarchical clustering algorithm, which is called single linkage. In a direct way, the MST conveys the hierarchical organization of the investigated assets and style factors, see Mantegna (1999)

for early applications in equity universes. This results in a tree structure as represented by the dendrogram in Figure 9.2. Moving up the tree, objects that are similar to each other are combined into branches, i.e. the higher the height of the fusion, the less similar the objects are.

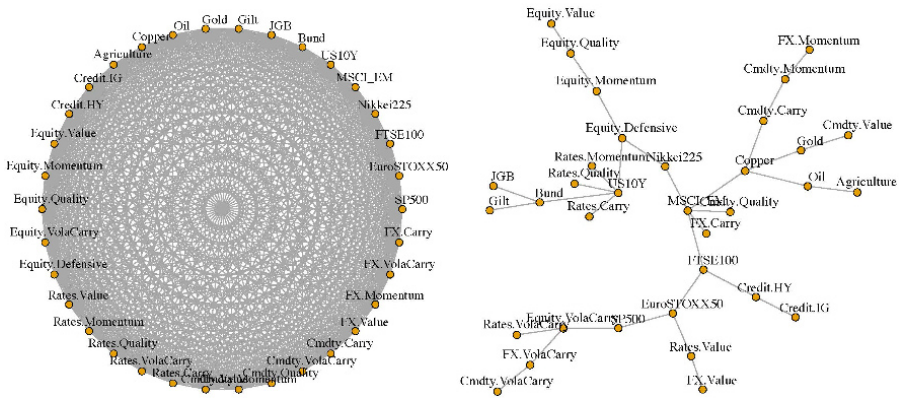


Figure 9.1. *Complete graph (left) versus minimum spanning tree (right)*

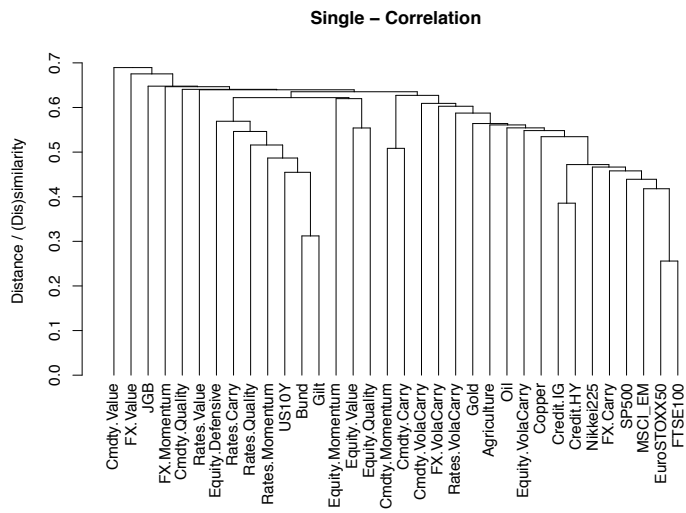


Figure 9.2. *Dendrogram (single linkage) associated with MST from Figure 9.1*

Another way of visualizing the embedded hierarchical structure is comparing the "unorganized" original correlation matrix to the correlation matrix resulting from reordering columns and rows according to the derived dendrogram, see Figure 9.5. This reordering results in a block-diagonal organization of the covariance matrix such that similar assets and style factors are placed together, whereas dissimilar assets (or factors) are placed further apart. This feature certainly seems helpful for discerning meaningful building blocks to diversify portfolios.

Algorithm 9.1. Recursive Bisection

1) Initialization
   a) Set the list of items: $L = \{L_0\}$, with $L_0 = \{n\}_{n=1,...,N}$ according to the order obtained by the dendrogram
   b) Assign a unit weight to all assets: $\omega_n = 1,\ \forall n = 1, ..., N$
2) If $|L_i| = 1,\ \forall L_i \in L$, then stop
3) For each $L_i \in L$ such that $|L_i| > 1$:
   a) Bisect L_i into two subsets, $L_i^{(1)} \cup L_i^{(2)} = L_i$, where $|L_i^{(1)}| = \text{round}\left(\frac{1}{2}|L_i|\right)$, and the order is preserved
   b) Define the variance of $L_i^{(j)}$, $j = 1, 2$ as $\tilde{V}_i^{(j)} = (\tilde{\omega}_i^{(j)})^T V_i^{(j)} \tilde{\omega}_i^{(j)}$, where $V_i^{(j)}$ is the covariance matrix between the constituents of the $L_i^{(j)}$ bisection and $\tilde{\omega}_i^{(j)}$ are the inverse variance weights
   c) Compute the split factor: $\alpha_i = 1 - \frac{\tilde{V}_i^{(1)}}{\tilde{V}_i^{(1)} + \tilde{V}_i^{(2)}}$
   d) Re-scale allocations ω_n by a factor of α_i, $\forall n \in L_i^{(1)}$
   e) Re-scale allocations ω_n by a factor of $(1 - \alpha_i)$, $\forall n \in L_i^{(1)}$
4) Loop to 2

Using the ensuing order of the assets, Lopez de Prado (2016) suggests an allocation technique based on recursive bisection and inverse variance allocation. Starting with one set of ordered assets and unit weights, he bisects the sets in each step and re-scales the weights of the assets in the resulting subsets with a factor α and $(1 - \alpha)$, respectively. The factor α is obtained by calculating the inverse variance allocation of the two subsets, where the variance of the subsets is determined by also using an inverse variance allocation within the subset. The procedure is summarized in Algorithm 9.1.

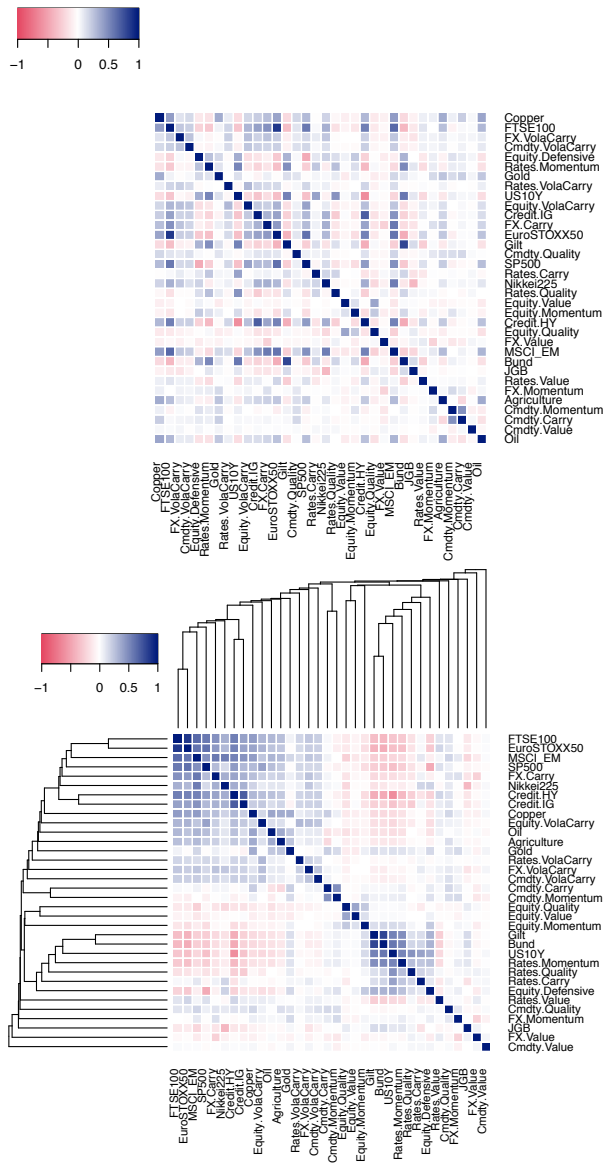


Figure 9.3. *"Random" correlation matrix versus quasi-diagonalized correlation matrix. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The introduced methodology of allocating according to hierarchical clustering is very flexible and we can conceive many variations thereof. For instance, various dissimilarity measures and hierarchical clustering methods can be used. Different functions for obtaining $\tilde{\omega}$ and α in steps $3.b)$ and $3.c)$ of Algorithm 9.1. can also be employed. Yet, by using recursive bisection, Lopez de Prado ignores the structure of the clusters at the different levels of the dendrogram and instead focuses on the order obtained at the bottom level. As this introduces a certain arbitrariness to the allocation algorithm, we will advance Algorithm 9.1. in a way that would split the allocations using the clusters obtained from the dendrogram.

9.1.3. *Hierarchical clustering*

Next, we will formally introduce hierarchical clustering by defining the dissimilarity measure and different linkage methods. We will also state an algorithm for obtaining the portfolio weights based on the dendrogram. We will assume that X is a $T \times N$ data matrix containing T observed returns for each of the N assets. X_j will denote the return times series of asset j.

Cluster analysis aims to group objects in clusters such that objects in a given cluster tend to be similar and objects in different clusters tend to be dissimilar. In particular, hierarchical clustering algorithms find recursively nested clusters, resulting in a tree-based representation called a dendrogram. Agglomerative clustering, also known as AGNES (agglomerative nesting), is the most commonly used hierarchical clustering method. Starting with each object being a single-element cluster, the algorithm works in a bottom-up manner, i.e. it sequentially merges the two most similar clusters until a single cluster contains all objects.

The notion of (dis)similarity between the considered objects is key for cluster analysis. Dissimilarity-based clustering algorithms, such as AGNES, require replacing the $T \times N$ data matrix X by an $N \times N$ dissimilarity matrix D. D is obtained by applying a dissimilarity measure to the data matrix X.

DEFINITION 9.1.– Dissimilarity measure. *Let X_i $i = 1, ..., N$ be the columns of a data matrix X, each containing T observations. A function $d : B \rightarrow$*

$[0, \infty)$ *is called the dissimilarity measure if*

$$d(X_i, X_j) \geq 0, \quad d(X_i, X_j) = d(X_j, X_i) \quad \textit{and}$$
$$d(X_i, X_i) = 0 \quad \forall i, j = 1, ...N, \tag{9.1}$$

where $B = \{(X_i, X_j)|i, j = 1, ..., N\}$.

Note that the dissimilarity matrix D defines a metric space, if function d additionally satisfies the triangle inequality. There are numerous ways of defining the dissimilarity measure, including Euclidean and Manhattan distances, but for the moment we will consider $d : B \to [0, 1]$ as defined by Lopez de Prado (2016):

$$d_{i,j} = d(X_i, X_j) = \sqrt{0.5(1 - \rho_{i,j})}, \tag{9.2}$$

where $\rho_{i,j} = \rho_{i,j}(X_i, X_j)$ is the Pearson correlation coefficient. We can verify that [9.2] is a dissimilarity measure, see, for instance, Lopez de Prado (2016). For perfectly correlated assets ($\rho_{i,j} = 1$), we have $d = 0$. For perfectly negatively correlated assets ($\rho_{i,j} = -1$), we have $d = 1$.

D contains the information about the dissimilarity between all single assets and factors. Still, we have to define how to use this information for measuring the (dis)similarity among clusters containing more than one element. This is done by the so-called linkage criterion. There are various criteria, but the most common ones are single linkage, complete linkage, average linkage and Ward's method, see, for instance, Raffinot (2017).

In the case of *single linkage*, the distance between two clusters is defined as the minimal distance of any two elements in the clusters:

$$d_{C_i, C_j} = \min_{x \in C_i, y \in C_j} d(x, y), \tag{9.3}$$

where C_i and C_j denote two clusters and x and y represent elements within the clusters. By construction, this linkage criterion is sensitive to outliers and can lead to long, straggly clusters, which is a well-known problem called chaining.

Complete linkage, in contrast, is defined as the maximum distance between any two elements of the clusters:

$$d_{C_i,C_j} = \max_{x \in C_i, y \in C_j} d(x,y). \quad [9.4]$$

The resulting clusters tend to be compact with a similar size. However, the strategy is also sensitive to outliers. Using *average linkage*, we compute the average of the distance of any two elements in the examined clusters:

$$d_{C_i,C_j} = \frac{1}{|C_i|}\frac{1}{|C_j|}\sum_{x \in C_i}\sum_{y \in C_j} d(x,y), \quad [9.5]$$

where $|C_i|$ is the cardinality of cluster i. *Ward's method* minimizes the total within-cluster variance:

$$d_{C_i,C_j} = \frac{|C_i||C_j|}{|C_i| + |C_j|}|\mu_i - \mu_j|^2, \quad [9.6]$$

where μ_i is the center of cluster i. This method aims to find compact clusters and is considered to be less sensitive to outliers.

Deriving the dendrogram, we start with each asset (or factor) in a single-element cluster. Then, based on the information provided by the dissimilarity matrix and the chosen linkage criterion, the two most similar clusters are sequentially merged. The dendrograms resulting from the four linkage methods can be observed in Figure 9.4: Ward's method results in compact clusters of similar size, making it a popular choice among researchers. Conversely, single linkage suffers from chaining, and the other methods are sensitive to outliers.

The dendrogram shows the merging of clusters in each step of the algorithm. Moving up the dendrogram, similar objects/clusters are combined into branches. On the axis, we provide the value at which the merge occurs, representing the (dis)similarity between two objects/clusters. However, we can only draw conclusions about the proximity of two objects/clusters if they are merged. The dendrogram does not allow for inferences about the proximity of two "random" objects/clusters.

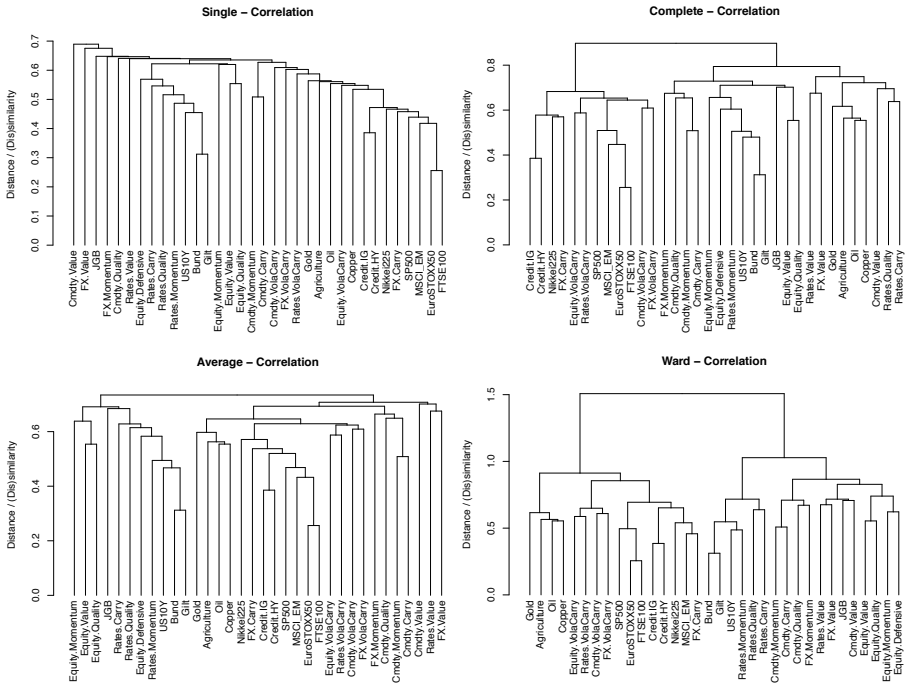


Figure 9.4. *Comparison of linkage methods: single linkage (top left), complete linkage (top right), average linkage (bottom left) and Ward's method (bottom right)*

In the case of a large-investment universe, the dendrogram might want to be considered only up to a certain level, instead of taking into account the whole hierarchical structure. While this reduction leads to a loss of information, finding the weight allocation is faster. Cutting the dendrogram will partition the assets and style factors into clusters. There are different ways to determine an optimal number of clusters. We could simply choose a plausible number by looking at the dendrogram or apply a statistical criterion for determining the "optimal" number of clusters. An example is given in Figure 9.5, where the number of clusters was deliberately chosen to be 7. Popular statistical methods are the gap statistic (Tibshirani *et al*. 2001) and the silhouette index (Rousseeuw 1987).

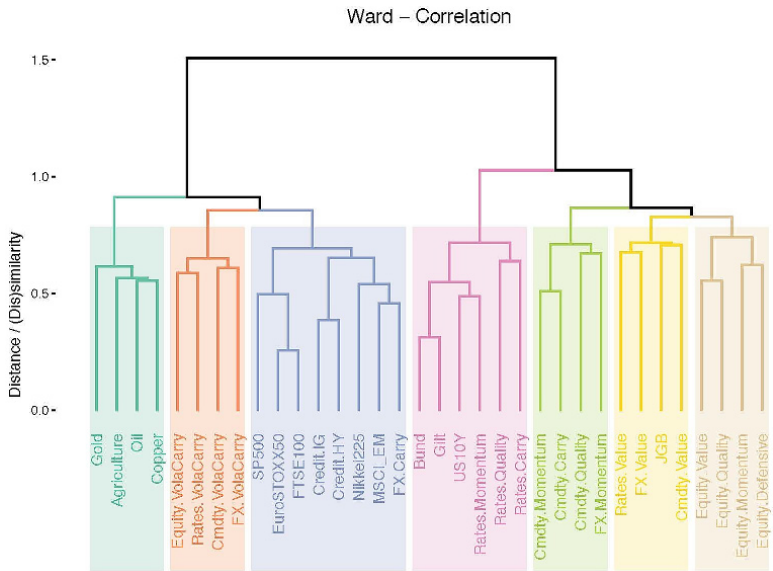


Figure 9.5. *Dendrogram based on Ward's method; cut such that seven clusters are obtained. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

9.1.4. *Portfolio allocation based on hierarchical clustering*

Having determined the dendrogram, we have to decide how to allocate capital. Instead of using an algorithm based on recursive bisection, we want to invest along the nodes of the dendrogram to acknowledge the hierarchical information. We also have to choose an allocation technique within and across clusters. Lopez de Prado uses the inverse variance strategy in both cases, yet there are various alternatives available. For instance, Papenbrock (2011) and Raffinot (2017) suggest a weighting scheme that allocates capital equally across the cluster hierarchy and within clusters. We will use different combinations of risk parity strategies, including inverse volatility and equal risk contribution. The general procedure is described in Algorithm 9.2., where we have to choose a methodology for the within- and across-cluster allocation. The algorithm starts at the top of the dendrogram and assigns weights by going from node to node. Note that Step 4 is only to be executed if an optimal number of clusters is used, i.e. not all remaining clusters are singleton clusters.

Algorithm 9.2. Clustering-based weight allocation

1) Perform hierarchical clustering and generate dendrogram
2) Assign all assets a unit weight $\omega_i = 1 \; \forall i = 1, ..., N$
3) For each dendrogram node (beginning from the top):
   a) Determine the members of clusters C_1 and C_2 belonging to the two sub-branches of the according dendrogram node
   b) Calculate the within cluster allocations $\tilde{\omega}_1$ and $\tilde{\omega}_2$ for C_1 and C_2 according to the chosen methodology
   c) Based on the within cluster allocations $\tilde{\omega}_1$ and $\tilde{\omega}_2$ calculate the across cluster allocation α (splitting factor) for C_1 and $(1 - \alpha)$ for C_2 according to the chosen methodology
   d) For each asset in C_1 re-scale allocation ω by factor α
   e) For each asset in C_2 re-scale allocation ω by factor $(1 - \alpha)$
4) For each cluster containing more than one element:
   a) Determine the members of the cluster
   b) Calculate the within cluster allocation
   c) For each asset in the cluster re-scale ω by the within cluster allocation
5) End

9.2. Tail dependency and hierarchical clustering

So far, we have investigated hierarchical clustering based on the correlation matrix. However, correlations can increase significantly during financial crises, because of contagion effects. Financial contagion can be triggered by unexpected extreme events and spread quickly. The best-known example is the Great Depression, which was triggered by the US stock market crash in 1929 and affected economies around the globe. Hence diversification by correlation clusters may fail to work when it is needed most.

Alternatively, we might group assets according to their co-movement when they experience large losses. Such an approach was proposed by De Luca and Zuccolotto (2011), who consider lower tail dependence coefficients (LTDC) to capture the pairwise tail behavior of the underlying assets. While the correlation coefficient describes the association of the entire distribution, the LTDC quantifies the association in extreme negative events. Presumably, lower tail dependence is more directly linked to the ability to diversify in crises, potentially improving the tail risk management of a given strategy.

9.2.1. *Tail dependence coefficients*

In the following, we formally introduce the lower and upper tail dependence coefficients and define a dissimilarity measure based on the LTDC. The main idea is to cluster assets that are characterized by a high probability of experiencing extremely negative events contemporaneously. The tail dependence coefficients are defined by a conditional probability: the probability that a random variable X takes an extreme value, given that an extreme value has occurred for random variable Y.

DEFINITION 9.2.– Tail dependence coefficients. *Let X and Y be two random variables and $F_X(x) = \mathbb{P}(X \leq x)$ and $F_Y(y) = \mathbb{P}(Y \leq y)$ be their distribution functions. The lower tail dependence coefficient (LTDC) is defined as*

$$\lambda_L = \lim_{t \searrow 0} \mathbb{P}(X \leq F_X^{-1}(t) | Y \leq F_Y^{-1}(t)) \quad [9.7]$$

and the upper tail dependence coefficient (UTDC) is

$$\lambda_U = \lim_{t \nearrow 1} \mathbb{P}(X > F_X^{-1}(t) | Y > F_Y^{-1}(t)). \quad [9.8]$$

We call X and Y asymptotically dependent (independent) in the lower tail if $0 < \lambda_L \leq 1$ ($\lambda_L = 0$), and accordingly for λ_U. Quantifying this probability solely depends on the assumed bivariate distribution, i.e. the existence of tail dependence is a property of the bivariate distribution, see De Luca and Zuccolotto (2011).

In applications, data is needed to estimate the tail dependence coefficients, based on a distributional assumption. The latter being rather difficult for financial data, copula functions are widely used instead. Copulas allow us to

describe the joint distribution function in terms of univariate marginal distribution functions. As a result, copulas allow us to effectively estimate the tail dependence in a simple and flexible manner. Assuming we can estimate the LTDC, we define the dissimilarity measure as suggested by De Luca and Zuccolotto (2011).

PROPOSITION 9.1.– *Let X and Y be two random variables and λ_L their associated lower tail dependence coefficient, then the following function, $d : \mathbb{R} \times \mathbb{R} \to [0, \infty)$, is a dissimilarity measure:*

$$d(X, Y) = -\log(\lambda_L). \qquad [9.9]$$

9.2.2. *Estimating tail dependence coefficients*

Copulas are popular in financial applications since they allow us to separate the analysis of a joint distribution function into two parts. First, the marginal distribution functions are analyzed independently for each random variable. Second, we choose a copula model to describe the dependence structure as a function of the margins or we use an empirical version of the copula. Hence a copula links the marginal distributions and the joint distribution and implicitly defines a bivariate distribution function.

One mainly distinguishes between parametric and non-parametric approaches for estimating copula-based tail coefficients. De Luca and Zuccolotto (2011), who first introduced a clustering approach based on the LTDC, used a parametric approach based on Archimedean copulas. Having selected a model and having estimated the according parameters, the LTDC can be easily computed. To avoid specifying a parametric model assumption on the pairwise dependence structure of the involved time series, we will briefly discuss two non-parametric approaches.

The procedure of Durante *et al*. (2015) is based on an extreme value theory approach that requires the definition of copulas, survival copulas and extreme value copulas. Moreover, we can establish a connection between these different concepts and the lower and upper tail dependence coefficients. Yet, in unreported results, we have found the dendrograms arising from this estimator to be rather unstable. As portfolio allocations along such dendrograms will suffer from high turnover, we are resorting to the more stable non-parametric estimator of Schmid and Schmidt (2007). The authors

propose a multi-dimensional generalization of the LTDC based on a conditional version of Spearman's rho, which weights the different parts of the multidimensional copula. We therefore refer to this estimator as the CSR (conditional Spearman's rho) estimator. For our purpose of estimating the pairwise LTDC, we are considering the two-dimensional version. See section 9.8 for the definition of the CSR estimator, and Schmid and Schmidt (2007) for its derivation.

Figure 9.6 shows a dendrogram based on [9.9], where the CSR estimator is used for estimating the LTDCs. In contrast to Figure 9.5, we can observe that the dissimilarity measures can lead to fairly different dendrograms. Note that we chose the optimal number of clusters to be five; other choices are conceivable. Intuitively, using the LTDC would see a joint cluster of commodity market and commodity style factors. Relative to the correlation-based dendrogram in Figure 9.5, we find that the equity-credit cluster is enlarged, including more style factors. Interestingly, equity style factors form one cluster, except for defensive equity, which clusters with the Japanese stock index.

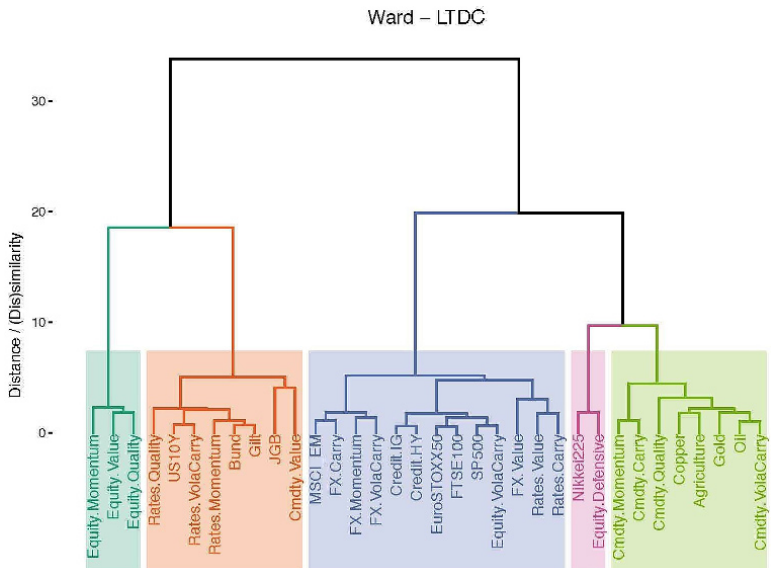


Figure 9.6. *Dendrogram (Ward method) - LTDC-based. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

In fact, there are various ways to define a dissimilarity measure based on the LTDC. Definition [9.9] is a popular choice for the following reasons: it ranges from 0 to infinity, since $\lambda_L \in [0, 1]$, it is also small when tail dependence is high and monotonically increases as tail dependence decreases. Hence, assets with high tail dependence appear to be "close" to each other, whereas assets with low tail dependence appear to be "more distant" from each other.

9.3. Risk-based allocation strategies

This section introduces risk-based allocation techniques. First, we focus on "traditional" allocation strategies, used to benchmark the hierarchical risk parity strategies. Second, we discuss Meucci's approach to managing diversification, which serves to construct a diversified risk parity strategy based on economic factors. The latter framework represents a useful benchmark to contrast with hierarchical risk parity strategies in terms of diversification.

9.3.1. *Classic risk-based allocation techniques*

The most simple allocation strategy is equal weighting $(1/N)$. It gives the same importance (in terms of dollars invested) to each asset. For N assets, we obtain weights

$$\omega_i = \frac{1}{N} \quad \text{for } i = 1, ..., N. \qquad [9.10]$$

Another popular risk-based allocation technique is the minimum-variance portfolio (MVP). It is a special case of the Markowitz mean-variance approach, determining the leftmost point on the efficient frontier. Its weights are determined by solving the quadratic optimization problem

$$\arg\min_{\omega} \frac{1}{2}\omega^T \Sigma \omega, \qquad [9.11]$$

where Σ is the covariance matrix. We impose full investment and long-only investment constraints. While minimum-variance portfolio weights are easy to compute, they often concentrate in low-volatility assets. Conversely, $1/N$ has a completely balanced weights allocation, but might suffer from a concentrated risk allocation, if the individual assets' risk differs significantly.

Therefore we may consider diversifying risk by allocating such that all assets contribute equally to the overall portfolio risk. Strategies following this objective are called risk parity strategies. A naive approach to risk parity would be to first, weight each asset inversely proportional to its volatility σ_i and, second, rescale weights (letting them add up to 1):

$$\omega_i = \frac{1/\sigma_i}{\sum_{j=1}^{N} 1/\sigma_j} \quad \text{for } i = 1, ..., N. \tag{9.12}$$

This inverse volatility strategy might be considered a naive risk parity strategy because it is agnostic with respect to asset correlations. Therefore we next discuss a risk parity strategy that assigns weights such that all assets truly contribute equally to total portfolio risk. A key metric is the marginal risk contribution, which measures the change in portfolio risk σ_p caused by an infinitesimal change in the weight of asset i:

$$MRC_i = \partial_{\omega_i} \sigma_p = \frac{(\Sigma\omega)_j}{\sqrt{w^T \Sigma w}}, \tag{9.13}$$

where Σ is the covariance matrix. The total risk contribution is then given by $TRC_i = w_i MRC_i$, and we can decompose the portfolio risk as follows

$$\sigma_p = \sum_{i=1}^{N} TRC_i. \tag{9.14}$$

Hence, a risk parity strategy with equal risk contributions must satisfy $TRC_i = TRC_j$ for all i, j. Since MRC_i and $(\Sigma\omega)_i$ are proportional, we might alternatively require $\omega_i(\Sigma\omega)_i = \omega_j(\Sigma\omega)_j$ for all i, j. However, there is no closed-form solution for the general case. We follow Maillard *et al.* (2010) and solve a numerical optimization problem in order to obtain the portfolio weights:

$$\arg\min_{\omega} \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\omega_i(\Sigma\omega)_i - \omega_j(\Sigma\omega)_j\right)^2. \tag{9.15}$$

9.3.2. *Diversified risk parity*

Striving for a well-diversified portfolio, Meucci (2009) constructs uncorrelated risk sources embedded in the underlying portfolio assets. A

well-diversified portfolio would follow a risk parity strategy applied to these uncorrelated risk sources. To measure the average number of uncorrelated bets in a portfolio, he introduces the concept of an effective number of bets. For constructing uncorrelated risk sources, Meucci (2009) suggests using principal component analysis (PCA), while follow-up research of Meucci *et al*. (2015) advocates a minimum-torsion transformation to derive the linear orthogonal transformation that is closest to the original assets (or a pre-specified factor model). In the following, we will consider this framework in the context of a factor model (rather than individual assets).

9.3.2.1. *Effective number of bets*

We are considering k correlated factors with return vector F. Thus, the portfolio return is given by

$$R_p = \sum_{i=1}^{k} b_i F_i, \qquad [9.16]$$

where b_i represents the portfolio weight of the i-th factor. Assuming that the portfolio can be expressed as a combination of uncorrelated factors $\tilde{F}$

$$R_p = \sum_{i=1}^{k} \tilde{b}_i \tilde{F}_i, \qquad [9.17]$$

we can compute the relative contribution of each uncorrelated factor to total risk:

$$p_i = \frac{Var(\tilde{b}_i \tilde{F}_i)}{Var(R_p)} = \frac{\tilde{b}_i^2 Var(\tilde{F}_i)}{\sum_{i=1}^{k} \tilde{b}_i^2 Var(\tilde{F}_i)}, \quad i = 1, ..., k. \qquad [9.18]$$

Note that the p_i sum to 1 and are non-negative. Hence, Meucci calls [9.18] the diversification distribution. He conceives a portfolio to be well-diversified if the diversification distribution is uniform, and uniformity is maximal for a risk parity strategy along the uncorrelated bets. To assess the degree of uniformity, Meucci defines the effective number of bets as the exponential of the negative entropy

$$N_{Ent} = \exp\left(-\sum_{i=1}^{k} p_i \ln(p_i) \right). \qquad [9.19]$$

To foster intuition, consider two extreme cases. A portfolio concentrated in a single factor leads to $p_i = 1$ for one i and $p_j = 0$ for $i \neq j$, resulting in N_{Ent}=1, a single bet. Vice versa, a well-diversified portfolio with $p_i = \frac{1}{N}$ for all i gives $N_{Ent} = k$, the maximum number of uncorrelated bets.

Meucci *et al*. (2015) give an explicit expression for the effective number of bets, letting t be a linear transformation such that

$$\tilde{F} = tF, \quad \tilde{b} = (t^T)^{-1}b, \tag{9.20}$$

where the bets $\tilde{F}$ are uncorrelated, the authors show the diversification distribution to be given by

$$p = \frac{((t^T)^{-1}) \circ (t\Sigma b)}{b^T \Sigma b}. \tag{9.21}$$

Lohre *et al*. (2014) denote the portfolio resulting from maximizing [9.19] as a diversified risk parity (DRP) portfolio. For [9.19] to be maximal, the diversification distribution has to be uniform. Hence, the analytical optimal portfolio can be found by choosing $\tilde{b}$ in [9.18] such that $p_i = \frac{1}{k}$ for $i = 1, ..., k$. This holds if

$$\tilde{b}_i = \frac{1}{\sqrt{Var(\tilde{F}_i)}}, \quad i = 1, ..., k. \tag{9.22}$$

Using [9.20], the analytical optimal solution can be expressed in terms of the original correlated factors. This solution only holds if no investment constraints are applied. However, many asset managers are restricted to long-only investments, in which case the following optimization has to be solved numerically

$$\omega_{DRP} = \arg\max_{b \in C} N_{Ent}(b), \tag{9.23}$$

where C is a set of portfolio constraints, such as long-only or full investment constraints.

9.3.2.2. *Principal component versus minimum-torsion bets*

While theoretically appealing, the PCA approach to construct uncorrelated bets (Meucci 2009) is subject to some criticism. Meucci *et al.* (2015) stress that the approach is purely statistical and might have little relation to the investment process. The ensuing principal portfolios can be difficult to interpret from an economic point of view and are not uniquely determined. Assuming e_k is an eigenvector, $-e_k$ is also an eigenvector. As a result, there are 2^k possible choices for constructing a maximally diversified portfolio. From a practical perspective, this is a serious concern, since using e_k or $-e_k$ translates to buying or selling a given principal portfolio. In addition, Meucci *et al.* (2015) point out that the principal portfolios are not invariant under simple scale transformation. Finally, the likely instability of the PCA is expected to translate into unduly high turnover in a related portfolio strategy.

To overcome the drawbacks of principal portfolios, Meucci *et al.* (2015) suggest another transformation to de-correlate the original factors F. Among all transformations t leading to uncorrelated factors, they choose the transformation $\tilde{t}$ that minimizes the tracking error with respect to the original factors:

$$\tilde{t}_{MT} = \arg \min_{Cr(tF)=I} NTE\{tF|F\} \quad [9.24]$$

where $Cr()$ denotes the correlation, I the identity matrix and NTE the multi-entry normalized tracking error

$$NTE(Z|F) = \sqrt{\frac{1}{k}\sum_k Var\left(\frac{Z_k - F_k}{\sigma(F_k)}\right)}. \quad [9.25]$$

Meucci *et al.* (2015) call this transformation the minimum-torsion transformation. The returns of the minimum-torsion portfolios are given by

$$\tilde{F}_{MT} = \tilde{t}_{MT}F, \quad [9.26]$$

where the subscript MT refers to minimum torsion. Meucci *et al.* (2015) provide an algorithm for solving [9.24]. Expressing the portfolio return in

terms of the uncorrelated minimum-torsion portfolios and using [9.26] leads to:

$$R_p = F^T b = \left(\tilde{t}_{MT}^{-1}\tilde{F}_{MT}\right)^T b = \tilde{F}_{MT}^T \left(\tilde{t}_{MT}^T\right)^{-1} b = \tilde{F}_{MT}^T \tilde{b}_{MT},$$

where $\tilde{b}_{MT} = \left(\tilde{t}_{MT}^T\right)^{-1} b$ are the minimum-torsion exposures (weights), which allow us to compute the diversification distribution and the effective number of bets via [9.21]:

$$\begin{aligned} p_{MT}(b) &= \frac{\left(\left(\tilde{t}_{MT}^T\right)^{-1} b\right) \circ \left(\tilde{t}_{MT}\Sigma_F b\right)}{b^T \Sigma_F b}, \quad N_{ent}(b) \\ &= \exp\left[-p_{MT}^T \ln(p_{MT}(b))\right]. \end{aligned} \qquad [9.27]$$

The minimum-torsion approach addresses all concerns regarding principal portfolios. The minimum-torsion portfolios are invariant under a scaling transformation. Moreover, the resulting factors are more stable and have an economic interpretation, since they are designed to be close to the original factors. For the same reason, the buy or sell decision of a given minimum-torsion portfolio is clearly defined.

9.4. Hierarchical risk parity for multi-asset multi-factor allocations

In this section, we focus on examining hierarchical risk parity strategies in the multi-asset multi-factor domain vis-à-vis the alternative risk-based allocation strategies. When it comes to performance comparisons, we make use of various performance measures, including Sharpe ratios and measures that focus on the downside risk potentially plaguing such factor strategies. In evaluating these strategies, we are mindful that their performance is just one realization over the historical path. Therefore, we resort to generating many alternative paths based on block-bootstrapping the historical asset and factor history. We will specifically use the stationary block-bootstrap of Politis and Romano (1994) with an expected block length of 15 days. This choice helps to preserve stylized facts of asset and factor returns, such as serial correlation and heteroskedasticity.

9.4.1. *Strategy universe*

The traditional risk-based allocation strategies are, first directly applied to the single assets and factors, and, second, to the eight aggregated factors

resulting from the imposed risk model. These eight factors can be viewed as "economic" clusters, providing a benchmark for the "statistical" hierarchical clustering. As for HRP, the allocation strategies used either within or across clusters are risk parity, either based on inverse volatility (IVP) or equal risk contributions (ERC). For hierarchical clustering, we use Ward's method and the dissimilarity matrices, based on either the correlation matrix or the LTDCs. For comparison purposes, two versions of Lopez De Prado's HRP strategy are considered, both based on recursive bisection: first, we replicate the original strategy, using single linkage and inverse variance allocation, as described in Algorithm 9.1. Second, we consider a variation thereof, described in Algorithm 9.2, using Ward's method and IVP, enabling comparability with the original HRP strategies of Lopez de Prado. An overview of the considered strategies of Lopez de Prado can be found in Table 9.1.

Abb.	Description	Formula	Dissimilarity	Linkage	Allocation method	Within and across-cluster allocation
Panel A: Strategies based on single assets and factors						
Equal	Equal weighted	[9.10]				
MVP	Minimum variance	[9.11]				
IVP	Inverse volatility	[9.12]				
ERC	Equal risk contribution	[9.15]				
Panel B: Strategies based on eight economic factors						
Equal_F	Equal weighted	[9.10]				
MVP_F	Minimum variance	[9.11]				
IVP_F	Inverse volatility	[9.12]				
ERC_F	Equal risk contribution	[9.15]				
DRP	Diversified risk parity	[9.22]				
Panel C: Strategies based on clusters						
DeP_Var	Hierarchical risk parity	Algorithm 9.1.	Correlation	Single	Bisection	Inverse variance
DeP_IVP	Hierarchical risk parity	Algorithm 9.1.	Correlation	Ward	Bisection	Inverse volatility
H_IVP_C	Hierarchical risk parity	Algorithm 9.2.	Correlation	Ward	Cluster	Inverse volatility
H_ERC_C	Hierarchical risk parity	Algorithm 9.2.	Correlation	Ward	Cluster	Equal risk contribution
H_IVP_L	Hierarchical risk parity	Algorithm 9.2.	LTDC	Ward	Cluster	Inverse volatility
H_ERC_L	Hierarchical risk parity	Algorithm 9.2.	LTDC	Ward	Cluster	Equal risk contribution

Table 9.1. *Overview of allocation strategies*

Portfolio rebalancing is conducted on a monthly basis. The strategies are assumed to be implemented using futures and swaps with associated transaction costs of 10 basis points and 35 basis points, respectively. Furthermore, eight basis points per month are considered for holding a given swap.

A two-year rolling window of daily returns is used for the estimation of the covariance matrix, and the resulting correlation-based dendrograms are updated every month. As suggested by De Luca and Zuccolotto (2011), we are using an eight-year rolling window to obtain an accurate estimate for the LTDCs. In addition, a univariate Student-t AR(1)-GARCH(1,1) model is applied to each time series to remove autocorrelation and heteroskedasticity. The resulting standardized residuals serve to obtain the pseudo-observations, required for the LTDC estimators.

9.4.2. *A statistical horse race of risk-based allocation strategies*

We performed backtests of the investment strategies in the six-year period from January 2012 to December 2017. This rather short sample period resonates with the eight-year estimation window required to determine the LTDCs. To foster intuition regarding the strategies' mechanics, we investigate weight and risk allocations for a subset of the discussed strategies, and Figure 9.7 shows their weight allocations (left) and volatility contributions by minimum-torsion portfolios (right). The MVP has a fairly high portfolio concentration, allocating up to 20% in individual assets and factors. In the case of the ERC strategy, all assets contribute equally to portfolio variance by design; nonetheless, the risk decomposition by minimum-torsion factors looks fairly balanced as well. Finally, the DRP that is designed to maximize diversification with regard to these minimum-torsion factors naturally shows a perfectly balanced risk allocation.

Turning to the hierarchical risk parity strategies in Figure 9.8, we find the original strategy of Lopez de Prado (2016), labeled $DeP\ IVP$, and the other correlation-based HRP strategy ($H\ IVP\ C$) to have a higher turnover than the traditional risk parity strategies. Notably, this turnover pattern is less pronounced for the HRP strategy based on lower tail dependencies, suggesting that the corresponding structure is more stable than the one based on correlations. All three HRP strategies exhibit certain concentrations in terms of the minimum-torsion portfolios, rendering them slightly less diversified than the traditional alternatives.

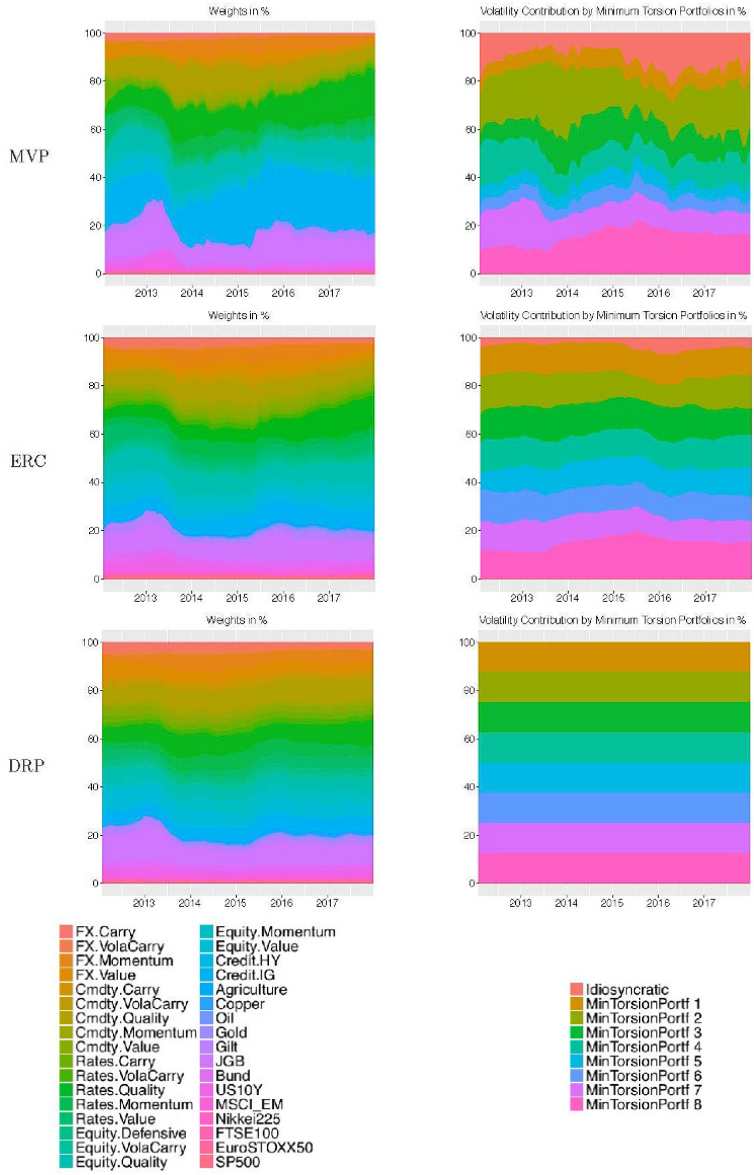


Figure 9.7. *Weight and risk decomposition of selected risk-based allocation strategies. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

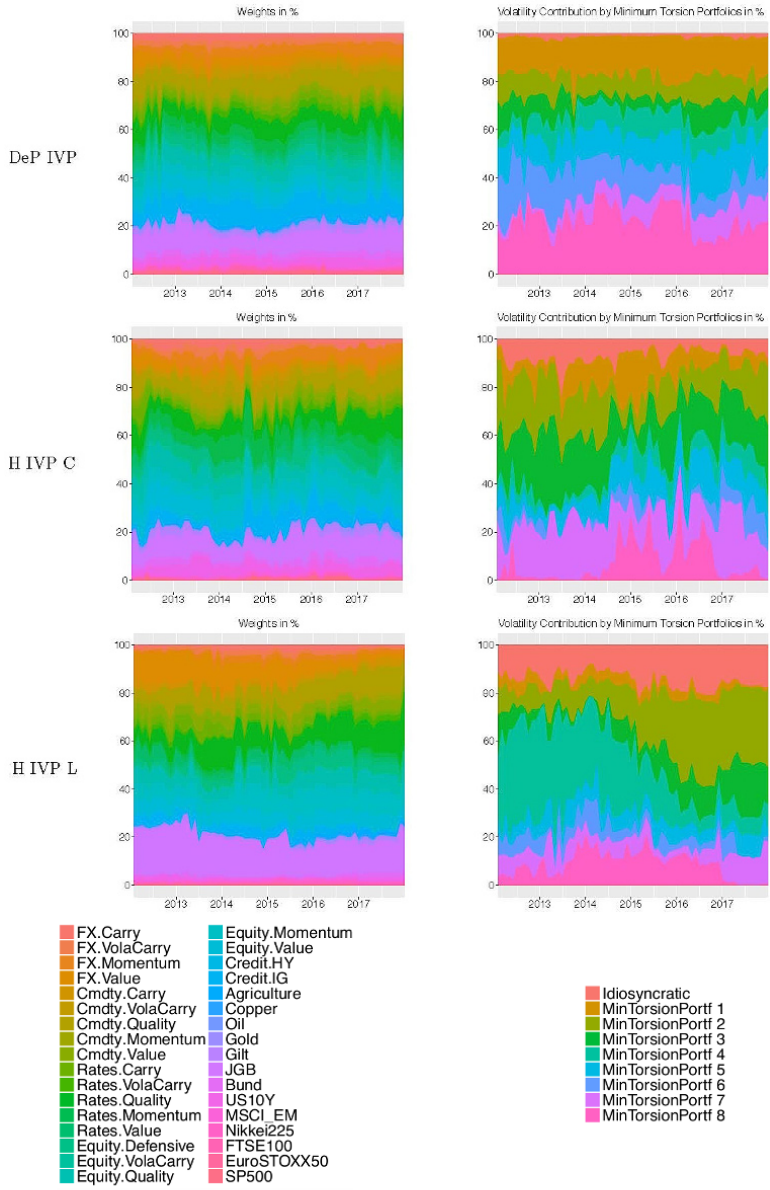


Figure 9.8. *Weight and risk decomposition of hierarchical risk parity strategies. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Table 9.2 shows performance and risk statistics as well as the average strategy turnover. First, we note that the $1/N$ strategy across single assets and factors has the highest return across all strategies (see Panel A). At the same time, $1/N$ suffers from the highest volatility as well as the highest maximum drawdown, rendering its risk-adjusted performance subpar. Notably, the underlying lack of diversification is not mitigated when considering economic factors as opposed to single assets and factors (Panel B); both variants hover around 3.5 bets averaged over time. Interestingly, minimum-variance optimization enables us to already double this number to 7.2 (or 6.7 for the MVP-variant based on economic factors). Unsurprisingly, these two portfolios exhibit the smallest portfolio volatilities in the sample period (0.84% and 0.90%, respectively). Of course, maximum drawdown figures and risk-adjusted returns are also improved relative to equal weighting.

Next, we examine the middle-ground solution, in between $1/N$ and minimum-variance: risk parity. In that regard, inverse volatility (ignoring correlations) for single assets and factors shows the second highest return, yet comes at the cost of some diversification. With 6.06 bets over time, it is short around two bets relative to the ERC variant (accounting for correlations). Still, risk-adjusted performance is fairly comparable across the two strategies, and their characteristics are more aligned when switching to economic factors as building blocks. Actually, the corresponding strategy, ERC_F, has fairly similar performance characteristics to the diversified risk parity (DRP) strategy, which is designed to have a maximum number of eight bets throughout time.

Having investigated the risk-based strategies for economic factors, we are eager to learn how the approaches based on statistical clusters fare. We start off by examining the original strategy of Lopez de Prado (2016). Its gross return is slightly higher than that of the risk parity variants, yet its turnover is more than three times higher ($\sim 18\%$ versus $\sim 5\%$). As a result, net returns are considerably smaller, as are net Sharpe ratios (0.94 versus 1.26 for ERC_F). Interestingly, the two further correlation-based HRP allocations (H_IVP_C and H_ERC_C) have even higher turnover (23.21% and 21.93%, respectively), bringing net Sharpe ratios down to 0.50 and 0.55.

	Risk-based allocations (assets)					
	Equal	MVP	IVP	ERC		
Gross return p.a. in %	3.45	2.37	2.79	2.67		
Net return p.a. in %	2.96	1.61	2.2	1.99		
Volatility p.a. in %	2.84	0.84	1.19	1.00		
Sharpe ratio	0.86	1.29	1.41	1.46		
Max drawdown in %	–5.30	–1.62	–1.89	–1.81		
Calmar ratio	0.53	0.94	1.12	1.05		
Sortino ratio	1.46	2.83	2.60	2.86		
# of Bets (Avg.)	3.75	7.19	6.06	7.90		
Turnover in % (avg.)	3.14	11.08	3.34	5.58		
	Risk-based allocations (economic factors)					
	Equal_F	MVP_F	IVP_F	ERC_F	DRP	
Gross return p.a. in %	2.13	2.32	2.41	2.51	2.57	
Net return p.a. in %	1.59	1.53	1.73	1.83	1.90	
Volatility p.a. in %	1.94	0.95	1.04	1.01	1.00	
Sharpe ratio	0.56	1.07	1.16	1.30	1.38	
Max drawdown in %	–4.00	–1.44	–1.51	–1.55	–1.56	
Calmar ratio	0.34	1.00	1.09	1.12	1.16	
Sortino ratio	1.16	2.35	2.38	2.62	2.74	
# of Bets (avg.)	3.54	6.65	7.42	7.95	8.00	
Turnover in % (avg.)	3.50	5.87	3.65	4.33	4.42	
	Correlation-based HRP allocations				LTDC-based HRP allocations	
	DeP_Var	DeP_IVP	H_IVP_C	H_ERC_C	H_IVP_L	H_ERC_L
Gross return p.a. in %	2.58	2.56	2.29	2.28	2.75	2.71
Net return p.a. in %	1.61	1.55	1.08	1.11	1.71	1.73
Volatility p.a. in %	1.17	1.10	1.14	1.10	1.10	1.08
Sharpe ratio	0.94	0.94	0.50	0.55	1.08	1.13
Max drawdown in %	–1.84	–1.65	–1.89	–1.83	–1.20	–1.29
Calmar ratio	0.82	0.89	0.47	0.52	1.68	1.48
Sortino ratio	1.91	1.98	1.31	1.40	2.68	2.67
# of Bets (avg.)	5.94	7.12	6.08	6.71	5.67	6.17
Turnover in % (avg.)	17.68	18.17	23.21	21.93	15.84	14.14

Table 9.2. *Performance and risk statistics for the multi-asset multi-factor case. All performance figures refer to net returns, except for the gross return. The sample period is from January 2012 to December 2017*

Finally, we investigate the effect of replacing the correlation-based dissimilarity matrix by one that is driven by LTDCs. We would hope for improved tail risk statistics of related strategies, and, indeed, we observe H_IVP_L and H_ERC_L to experience the two smallest maximum drawdowns over the sample period, –1.20% and –1.29% respectively. These compare to the third smallest value for MVP_F (–1.44%) and the fourth smallest value for IVP_F (–1.51%). The two LTDC-based HRP strategies' turnover is less elevated, rendering these two strategies more competitive relative to the alternative risk parity strategies in terms of (risk-adjusted) returns.

Of course, all of the above evidence might be considered anecdotal, as they merely refer to the realization over the historical path (which is also rather short). We wish to foster intuition with regard to the distributional properties of the obtained performance statistics. Using the 2003–2017 data history, we generated 2000 "alternative" six-year investment periods based on the stationary block-bootstrap of Politis and Romano (1994) with an average block length of 15 days[2]. Figure 9.9 shows the boxplots of the associated performance and risk statistics. Note that we have dropped Equal and Equal_F for not distorting the charts.

By and large, we find the block-bootstrapped results to be consistent with the historical evidence. Still, it seems that the correlation-based HRP strategies have higher median net returns, even though their turnover is higher than what was historically observed. The corresponding dendrograms' instability is crucially driving the high turnover. In total, the general return ranking is unchanged and we note that strategies based on single assets and factors have wider return distributions than those based on economic factors. These findings also carry over to net Sharpe ratios. A further divergence from the historical evidence is that the median maximum drawdowns of the HRP strategies based on LTDCs are much lower than the maximum drawdowns obtained in the backtest, ultimately putting them on a par with the maximum

2 Given the computational effort of fitting the AR-GARCH models, the LTDC-based dendrogram is only updated once a year for the purpose of the block-bootstrap analysis. This yearly updating of the dendrogram reduces the strategy turnover considerably, yet it likewise reduces the strategy's ability to adjust timely to changes in the risk environment. These two effects tend to, on average, cancel out, and unreported results document the corresponding historical backtest to be consistent in terms of net returns with that based on monthly updating.

drawdown results of strategies based on economic factors. Hence, the observed backtest advantage might be deemed more apparent than real. Against this backdrop, the ranking of downside risk-adjusted returns (Calmar ratio) are aligned more along strategies either operating across economic factors or statistical clusters.

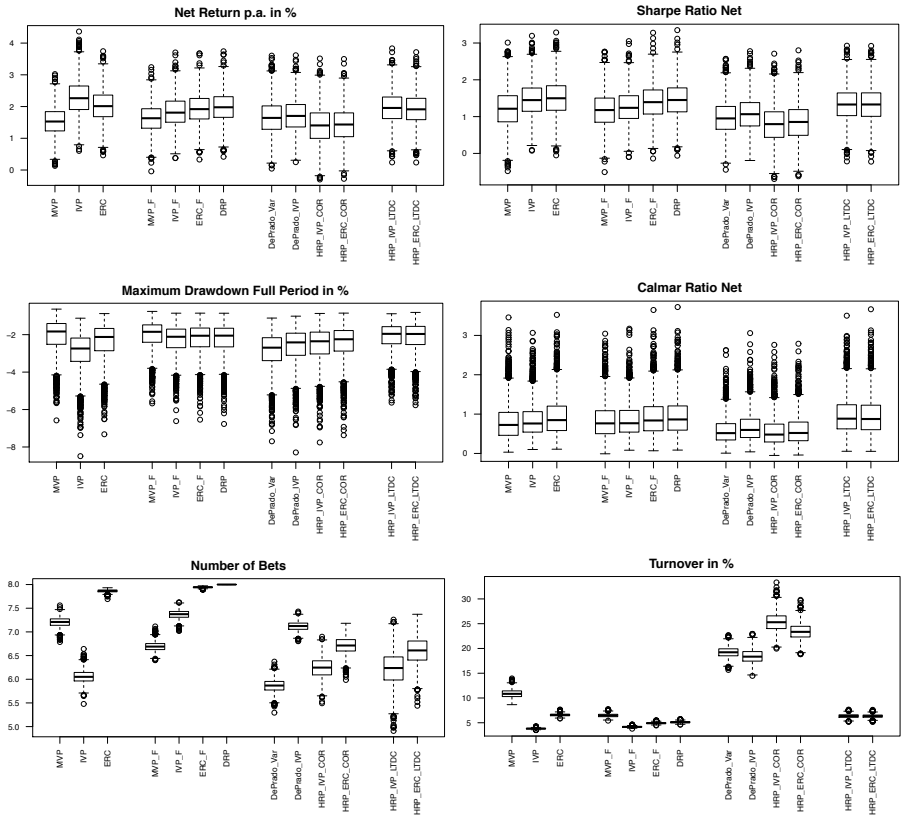


Figure 9.9. *Boxplots of performance and risk statistics for a six-year investment horizon, based on 2000 block-bootstrap simulations*

9.5. Conclusion

Building upon the hierarchical risk parity of Lopez de Prado (2016), we proposed and investigated innovative variations of this strategy. The main motivation to base an allocation strategy on hierarchical clustering is that the

correlation matrix is too complex to be fully analyzed and lacks the notion of hierarchy. Hierarchical clustering allows us to reduce the associated complete graph of information into a minimum spanning tree, thus focusing on the correlations that really matter. Hierarchical risk parity is an intuitive investment approach, allowing for a high degree of flexibility. We can choose from various dissimilarity measures, trying to capture different properties and relations among the underlying return time series. Moreover, we can modify the resulting tree structure by considering various linkage methods. The only restriction is imposing that weights of within- and across-cluster allocation strategies each sum to 1.

As an alternative to the common choice of the Pearson correlation coefficient, we build a dissimilarity measure based on the lower tail dependence to explicitly incorporate the notion of tail risk management. Such innovation is expected to help navigate the multi-asset multi-factor universe that comes with significant tail risk for some of its constituents. Having obtained the hierarchical structure of assets and factors in terms of the dendrogram, a simple and efficient capital allocation is applied within and across clusters. Lopez de Prado (2016) suggested using an algorithm based on recursive bisection, which only considers the order of the assets at the "bottom" of the dendrogram and ignores the nested structure of the clusters. Conversely, we defined an allocation algorithm that fully uses the hierarchical structure by allocating capital in a top-to-bottom fashion according to the clusters created at each level of the dendrogram.

A multi-asset multi-factor case study suggests that hierarchical risk parity strategies have the potential to compete with traditional risk parity strategies. Moreover, they can add desirable diversification properties. The hierarchical risk parity strategies based on the lower tail dependence coefficient especially seem to benefit tail risk management. Notably, traditional risk parity strategies, based on economic factors, can be expected to provide downside risk protection that is on a par with these HRP strategies. Thus diversifying across economic factor clusters is also meaningful. Yet higher-dimensional investment universes could make the "economic" clustering less intuitive and thus more difficult to come up with reasonable clusters. In contrast, ignoring computational intensity, hierarchical clustering methods can be directly applied to high-dimensional investment universes.

9.6. Acknowledgements

We are grateful to Victoria Beschorner, Emmanuel Jurczenko, Erhard Radatz, Lukasz Szpruch and seminar participants at the 2018 Global Research Meeting of Invesco Quantitative Strategies in Frankfurt. Note that this paper expresses the authors' views, which do not necessarily coincide with those of Invesco or Metzler.

9.7. Appendix 1: Definition of style factors

9.7.1. *Foreign exchange (FX) style factors*

Data	Style	Description
GS	Carry	The FX Carry strategy has historically benefited from the tendency of FX forwards of high-yielding currencies to overestimate the actual depreciation of future FX Spot. On a monthly basis, the strategy evaluates the implied carry rate (FX Forward vs FX Spot) of a number of currencies (G10 and EM) against the USD, and ranks them based on that measure. The strategy goes long single-currency indices (which roll FX forwards) for the currencies with the highest carry and short single-currency indices for the currencies with the lowest carry.
GS	Value	The FX valuation strategy relies on exchange rates reverting back to their fair value in the medium- to long-term horizons. On a monthly basis, the strategy evaluates the valuation measure (based on GS DEER, dynamic equilibrium exchange rate model) of a number of currencies (G10 and EM) against the USD, and ranks them based on that measure. The strategy goes long single-currency indices (which roll FX forwards) for the currencies with the highest ranking (i.e. most undervalued) and short single-currency indices for the currencies with the lowest ranking (i.e. most overvalued).
GS	Momentum	This factor capitalizes on the persistence of trends in forward exchange rate movements which are driven by both carry and spot movements. On a daily basis, the strategy evaluates the recent performance of 27 currencies against the USD. It then takes either a long or short position on each of those currencies against the USD, depending on whether their actual performance has been positive or negative.
GS	Volatility Carry	On average, implied volatility tends to trade at a premium to subsequent realized volatility as investors demand a risk premium for selling optionality and being short volatility. This preference is driven by risk aversion, i.e. market participants attach a higher value to being long protection. FX Volatility Carry sells short-dated options (puts and calls) on several currencies and delta-hedges on a daily basis.

9.7.2. *Commodity style factors*

Data	Style	Description
GS	Carry	Captures tendency for commodities with tighter time spreads to outperform due to low inventories driving both backwardated futures curves and price appreciation and buying demand from consumer hedgers for protection against price spikes in undersupplied commodities. The strategy goes long the top third and short the bottom third of the 24 commodities from the S&P GSCI universe, ranked by annualized strength of front month time spreads. The strategy is rebalanced daily, based on the signals over the last 10 days. The strategy is net of cost.
GS	Value	The strategy uses the weekly Commodity Futures Trading Commission (CFTC) positioning data to determine whether to go long and short in commodities and will take long positions in the commodities that the cumulative positions are most short and short positions in the commodities that speculative positions are most long.
GS	Momentum	Momentum in commodity returns reflect initial underreaction or subsequent overreaction to changes in demand, as increasing or decreasing supply takes many years to implement and subsequently overshoot required changes to match demand. The strategy goes long the top third and short the bottom third of the 24 commodities from the S&P GSCI universe, ranked by rolling one-year excess returns of each commodity. The strategy is rebalanced daily based on the signals over the last 10 days. The strategy is net of cost.
GS	Quality	This factor captures the tendency for deferred futures contracts to outperform nearer dated contracts due to producers hedging further out than consumers and passive investors investing near the front of the curve. The strategy goes long selected points on the curve of each commodity, equally weighted amongst commodities. The strategy goes short an equally weighted basket of the nearest commodity contracts, beta-adjusted at the basket level.
GS	Volatility Carry	Options on commodities are used by hedgers and investors to protect against falling or rising prices or to speculate with limited downside risk. Such participants are generally buyers of implied volatility. Therefore, the strategy aims to capture the risk premium of convexity transfer.

9.7.3. *Rates style factors*

Data	Style	Description
GS	Momentum	This factor capitalizes on the persistence of trends in short- and long-term interest rate movements. On a daily basis, the strategy evaluates the recent performance of a number of futures contracts for the United States, Germany, Japan and the United Kingdom. It then takes either a long or short position on each of the futures, depending on whether their actual performance has been positive or negative.
GS	Quality	This factor capitalizes on the observation that risk-adjusted returns at the short end of the curve tend to be higher than at the long end. A leveraged long position on the former versus the latter tends to capture positive excess returns as compensation for the risk premium that stems from investors having leverage constraints and favoring long-term rates. The interest rates curve strategy enters a long position on 5y US Bond futures and a short position on 30y Bond futures, as well as a long position on 5y German Bond futures and a short position on 10y German Bond futures, rolling every quarter. The exposure to each future is adjusted to approximate a duration-neutral position.
GS	Volatility Carry	On average, Treasury Futures implied volatility tends to trade at a premium to subsequent realized volatility, as investors demand a risk premium for selling optionality and being short volatility. This preference is driven by risk aversion, i.e. market participants attach a higher value to being long protection. Interest Rates Volatility Carry sells short-dated options (puts and calls) monthly on 10y Treasury Futures and delta-hedges on a daily basis.

9.7.4. *Equity style factors*

Data	Style	Description
IQS	Value	The value factor refers to the finding that value stocks, characterized by valuation ratios, offer higher long-run average returns than growth stocks, characterized by high valuation ratios. To optimally combine factors in a multi-factor asset allocation exercise, we construct the factor to have zero exposure to other factors and thus does not alter the overall portfolio exposure to other factors when added to the portfolio.

IQS	Momentum	The momentum factor captures a medium-term continuation effect in returns by buying recent winners and selling recent losers. The factor combines price as well as earnings momentum information. To optimally combine factors in a multi-factor asset allocation exercise, we construct the factor to have zero exposure to other factors and thus does not alter the overall portfolio exposure to other factors when added to the portfolio.
IQS	Quality	The quality factor combines different measures of determining financial health and operating profitability. To optimally combine factors in a multi-factor asset allocation exercise, we construct the factor to have zero exposure to other factors and thus does not alter the overall portfolio exposure to other factors when added to the portfolio.
IQS	Defensive	The defensive factor refers to the finding that low volatility stocks tend to outperform high volatile stocks on a risk-adjusted basis. To capture this behavior, the factor is constructed to go long a minimum-variance portfolio and short the beta-portion of the market
GS	Volatility Carry	On average, equity implied volatility tends to trade at a premium to subsequent realized volatility as investors demand a risk premium for selling optionality and being short volatility. This preference is driven by risk aversion, i.e. market participants attach a higher value to being long protection. Global Equity Volatility Carry sells short-dated options (puts and calls) daily on global indexes and delta-hedges. It buys back lower puts to mitigate drawdowns without diluting the underlying alpha drivers.

9.8. Appendix 2: CSR estimator

We state the estimator as presented in Schmid and Schmidt (2007). Consider a random sample $(X_j)_{j=1,\ldots,n}$ from a d-dimensional vector X with joint distribution function F and copula C. Based on the multivariate conditional version of Spearman's rho (CSR), given by

$$\rho(p) = \frac{\int_{[0,p]^d} C(u)du - \left(\frac{p^2}{2}\right)^d}{\frac{p^{d+1}}{d+1} - \left(\frac{p^2}{2}\right)^d}, \quad 0 < p \leq 1, \qquad [9.28]$$

they define a multivariate LTDC ρ_L, a generalization of the bivariate LTDC λ_L, by

$$\rho_L = \lim_{p \searrow 0} \rho(p) = \lim_{p \searrow 0} \frac{d+1}{p^{d+1}} \int_{[0,p]^d} C(u)du. \quad [9.29]$$

Assuming that neither the univariate distribution functions F_{X_j} nor the according copula C are known, empirical versions have to be considered. An empirical estimator for the marginal distributions is given by:

$$\hat{F}_{i,n}(x) = \frac{1}{n} \sum_{j=1}^{n} \mathbb{1}_{\{X_{ij} \leq x\}}, \quad \text{for } i = 1, ..., d \text{ and } x \in \mathbb{R}. \quad [9.30]$$

Set $\hat{U}_{ij,n} = \hat{F}_{i,n}(X_{ij})$ for $i = 1, ..., d$, $j = 1, ..., n$ and $\hat{U}_{j,n} = (\hat{U}_{1j,n}, ..., \hat{U}_{dj,n})$. Note that

$$\hat{U}_{ij,n} = \frac{1}{n} \left(\text{rank of } X_{ij} \text{ in } X_{i1}, ..., X_{in}\right). \quad [9.31]$$

The copula C is estimated by the empirical copula:

$$\hat{C}_n(u) = \frac{1}{n} \sum_{j=1}^{n} \prod_{i=1}^{d} \mathbb{1}_{\{\hat{U}_{ij,n} \leq u_i\}} \quad \text{for } u = (u_1, ..., u_d)^T \in [0,1]^d, \quad [9.32]$$

leading to the following estimator for [9.28]:

$$\hat{\rho}_n(p) = \left\{ \frac{1}{n} \sum_{j=1}^{n} \prod_{i=1}^{d} \left(p - \hat{U}_{ij,n}\right)^+ - \left(\frac{p^2}{2}\right)^d \right\} \Bigg/ \left\{ \frac{p^{d+1}}{d+1} - \left(\frac{p^2}{2}\right)^d \right\}. \quad [9.33]$$

Finally, the estimator for [9.29] is given by:

$$\hat{\rho}_L = \hat{\rho}_n \left(\frac{k}{n}\right), \quad [9.34]$$

where $k \in \{1, ..., n\}$ has to be chosen by the statistician.

9.9. References

Bernardi, S., Leippold, M. and Lohre, H. (2018). Maximum diversification strategies along commodity risk factors. *European Financial Management*, 24(1), 53–78.

Boudt, K., Carl, P. and Peterson, B. (2013). Asset allocation with conditional value-at-risk budgets. *Journal of Risk*, 15(3), 39–68.

De Luca, G. and Zuccolotto, P. (2011). A tail dependence-based dissimilarity measure for financial time series clustering. *Advances in Data Analysis and Classification*, 5(4), 323–340.

Durante, F., Pappadà, R. and Torelli, N. (2015). Clustering of time series via non-parametric tail dependence estimation. *Statistical Papers*, 56(3), 701–721.

Jurczenko, E. and Teiletche, J. (2019). Expected shortfall asset allocation: A multi-dimensional risk budgeting framework. *Journal of Alternative Investments*, 22(Fall), 1–16.

Lohre, H., Opfer, H. and Ország, G. (2014). Diversifying risk parity. *Journal of Risk*, 16(5), 53–79.

Lopez de Prado, M. (2016). Building diversified portfolios that outperform out-of-sample. *Journal of Portfolio Management*, 42(4), 59–69.

Maillard, S., Roncalli, T. and Teiletche, J. (2010). The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, 36(4), 60–70.

Mantegna, R. (1999). Hierarchical structure in financial markets. *The European Physical Journal B – Condensed Matter and Complex Systems*, 11(1), 193–197.

Markowitz, H. (1952). Portfolio selection. *Journal of Finance*, 7(1), 77–91.

Meucci, A. (2009). Managing diversification. *Risk*, 35–40.

Meucci, A., Santangelo, A. and Deguest, R. (2015). Risk budgeting and diversification based on optimized uncorrelated factors. *Risk*, 11(29), 70–75.

Papenbrock, J. (2011). Asset clusters and asset networks in financial risk management and portfolio optimization. PhD Thesis, Karlsruhe Institute of Technology, Karlsruhe.

Politis, D.N. and Romano, J.P. (1994). The stationary bootstrap. *Journal of the American Statistical Association*, 89(428), 1303–1313.

Prim, R.C. (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6), 1389–1401.

Raffinot, T. (2017). Hierarchical clustering-based asset allocation. *Journal of Portfolio Management*, 44(2), 89–99.

Rousseeuw, P.J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(C), 53–65.

Schmid, F. and Schmidt, R. (2007). Multivariate conditional versions of Spearman's rho and related measures of tail dependence. *Journal of Multivariate Analysis*, 98(6), 1123–1140.

Tibshirani, R., Walther, G. and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2), 411–423.

Tumminello, M., Lillo, F. and Mantegna, R.N. (2010). Correlation, hierarchies, and networks in financial markets. *Journal of Economic Behavior and Organization*, 75(1), 40–58.

10

Portfolio Performance Attribution: A Machine Learning-Based Approach

Using a classification and regression tree-inspired machine learning approach, we conducted performance attribution of several factor-tilted portfolios in the US equity market. We found that this methodology was able to identify the systematic sources of return in such portfolios and explain the majority of the tracking error of such strategies. The algorithm detailed here offers significantly greater explanatory power than other more commonly used attribution approaches.

10.1. Introduction

The performance of investment portfolios is increasingly seen as a combination of return to systematic factors ("beta") and the active judgment of the underlying investment manager. Accurate performance attribution of an investment portfolio involves separation of systematic factors from the active management, so that the investor can distinguish the component of the return that is due purely to systematic factors from that which is due to the investor skill – a task that has substantial implications for the fee that is associated with the underlying strategy. Portfolios in which a substantial component of the returns are attributable to systematic factors are expected to have a correspondingly lower fee.

Separating the systematic component of returns is usually accomplished by either a returns-based analysis, a holdings-based analysis or, most often, a combination of the two. Returns-based analysis typically requires longer periods for evaluation – for example, three to five years of monthly returns.

Chapter written by Ryan BROWN, Harindra DE SILVA and Patrick D. NEAL.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Holdings-based analysis, using the methodology first presented by Brinson and Fachler (1985), can be applied over much shorter periods. This methodology, commonly known as "Brinson attribution", can be applied to any arbitrary time period, although it is most commonly applied to monthly holding periods. While applications of the framework have typically used countries and sectors as the underlying systematic factors, the basic framework can be easily adapted to any continuous variable by using discrete categorical variables at different levels of the continuous variable.

Two challenges are associated with Brinson-based attribution: first, the choice of cutoffs points for creating the categorical variables; second, the curse of dimensionality. For example, if we are analyzing the return contribution of a single continuous variable, the process requires the transformation of this variable into levels. Most commonly it is four or five, but sometimes it is as few as two and as many as 10 – to capture the variation in the return associated with this variable. The category cutoffs are usually based on an equal number of observations falling into each group, often weighted by market capitalization. When more than one variable is used, the interaction between the variables creates a large number of categories. For example, if we are analyzing the impact of market capitalization and earnings yield on portfolio returns, the use of five categories for each variable would result in the use of 25 (i.e. 5×5) categorical variables to capture the interaction between these factors. This is especially challenging when the number of stocks in the universe is small, as the correspondingly smaller number of stocks in each group will result in the groups' returns being dominated by idiosyncratic effects.

In this chapter, we use a machine learning-based approach to address both these issues. First, we base the choice of variable in each period, as well as the cutoff used to define groups with this variable, on their ability to explain the active return of a portfolio. Second, interactions between the variables are also empirically derived in terms of this same criterion. The approach is illustrated using several case studies, and we significantly reduce the level of unexplained active return. The process of minimizing the unexplained active return also serves as a good criterion by which to identify the systematic tilts in a factor-tilted portfolio.

The merits of this approach are that the automated selection of the variables that best explain the active returns of the portfolio in question enables the analysis of a large number of strategies. In addition the potential for improved identification of the level of systematic tilts increases the degree of economically meaningful performance attribution. This technique can easily scale across strategies, time periods and factor libraries.

A criticism of the approach used in this chapter is that the machine learning used here is not "out of sample", and therefore the result that it is superior to the naive attribution is not a surprise. However, the analysis presented focuses on the ability to *ex post* explain the active returns of a portfolio. The goal of the research presented here is to quantify the potential benefits of using machine learning in order to analyze active returns. Given the growing number of factor-tilted strategies, a potential application of this type of approach would be to quantify the components of the return that come from pure factor tilts, absent assumptions about a particular risk or pricing model, such as the Fama–French 4 factor model (Carhart 1997).

The following section introduces the notation and algorithm used in this chapter. This is followed by a description of the data and a discussion of the results stemming from the application of the process to common factor-based portfolio strategies that are currently used by investors. This chapter concludes with a summary of the results, as well as a discussion of potential applications of this approach to performance attribution.

10.2. Methodology

Our starting point is the Brinson–Fachler attribution model. This holdings-based performance attribution approach decomposes benchmark-relative returns into two effects. "Allocation effects" are interpreted as the manager's ability to overweight high-performing and underweight low-performing groups of stocks. The other category is termed "residual effects", which are interpreted as a manager's ability to identify winning and losing stocks within said groups.

Allocation effects for each category j are represented in Brinson–Fachler attribution by the following equation:

$$allocation_j = (w_j{}^p - w_j{}^b) * (r_j{}^b - R^b)$$

That is, for every (sector, country, factor quantile, etc.) category j, the allocation effect is the active weight in the category (the difference between the portfolio weight ${w_j}^p$ and the benchmark weight ${w_j}^b$) times the benchmark relative performance of category j (category return ${r_j}^b$ minus benchmark total return R^b). A fundamental requirement is that each security belongs to only one category. The total allocation effect is the sum of these group effects. Positive active weights in outperforming categories and negative active weights in underperforming categories both increase allocation effects.

Residual effects are defined as the remaining active return not captured by the allocation effect calculation[1]:

$$residual_j = {w_j}^p * ({r_j}^p - {r_j}^b).$$

The sum of both the allocation and residual effects across all groups is equal to the portfolio's active return.

10.2.1. *Matrix algebra representation of selection and allocation effects*

In order to explicitly identify the role of securities' category assignments in computing the magnitudes of allocation and residual effects, we write the Brinson attribution and the category returns using linear algebra. We first define three stock-level vectors necessary to perform a Brinson-style factor decomposition:

p_{n_1} = portfolio weights;

b_{n_1} = benchmark weights;

r_{n_1} = security returns.

We then choose a vector consisting of one explanatory continuous variable x_{n_1}. Given a classification based on x_{n_1} into J groups, we can transform this x vector into a matrix of classification variables: X_{n_j}, with

1 In many Brinson formulations, the "residual effect" is further decomposed into "selection" and "interaction" effects – defined as $({r_j}^p - {r_j}^b) * {w_j}^b$ and $({r_j}^p - {r_j}^b) * ({w_j}^p - {w_j}^b)$, respectively. We follow the accepted practice of combining these two nested effects.

the j^{th} column representing membership in that group. For example, if our explanatory variable is beta and we are performing analysis on four equal-sized groups (i.e. quartiles), then a row value of 1 in the first of four columns of X represents membership of that stock in the lowest quartile of beta.

With this representation, we can calculate the J groupwise exposures for the portfolio and benchmark as $p'X$ and $b'X$. Group active weights are defined analogously as $(p-b)'X$.

Another necessary intermediate step is to diagonalize the portfolio and benchmark weights. The $diag()$ operator creates a square matrix, with zeros on the off diagonal and the elements of the argument along the main diagonal:

$\Lambda^p = diag(p_{n\_1})$ = portfolio weights;

$\Lambda^b = diag(b_{n\_1})$ = benchmark weights.

Using this matrix notation, we can express the J group-decomposed portfolio and benchmark returns as:

$(X'\Lambda^p X)^{-1}X'\Lambda^p r$ = groupwise portfolio returns;

$(X'\Lambda^b X)^{-1}X'\Lambda^b r$ = groupwise benchmark returns;

This decomposition can be viewed in the context of estimated coefficients from a WLS regression, with $\{r, X, \Lambda\}$ as the $\{y, X, wt\}$ variables, respectively.

With these reformulations, the residual component of the Brinson decomposition can be written as the portfolio exposure multiplied by the active returns. The matrix formulation is:

$$X'p * \{(X'\Lambda^p X)^{-1}X'\Lambda^p r - (X'\Lambda^b X)^{-1}X'\Lambda^b r\},$$

where $*$ represents the multiplicative elementwise operator.

The sum of the residual components, which is a scalar, can be succinctly written as:

$$\begin{aligned} total\_residual\_effect &= TRE \\ &= p'X\{(X'\Lambda^p X)^{-1}X'\Lambda^p r - (X'\Lambda^b X)^{-1}X'\Lambda^b r\} \end{aligned}$$

In turn, the sum of all group allocation effects can be formulated as:

$$\begin{aligned} total\_allocation\_effect = TAE \\ = (p-b)'X\{(X'\Lambda^b X)^{-1}X'\Lambda^b r \\ - b'X(X'\Lambda^b X)^{-1}X'\Lambda^b r\} \end{aligned}$$

When expressed in this way, we can see that neither the residual nor the allocation effects are linear in portfolio weights, benchmark weights, security returns or factor categories. Furthermore, we see that failure to use the appropriate variables in estimating the allocation effects can result in inaccurate conclusions about the selection ability of a strategy.

Using this approach, the returns are evaluated over a single period. To perform multiperiod analysis, the returns are linked together, and there are several commonly used techniques to link different attribution components (see, for example, Menchero 2005). In our analysis, we choose not to link the different periods together and all multiperiod results are presented as simple arithmetic averages for ease of understanding. As others have noted (Frongello 2002), although the mathematics of linking differ greatly among the various methods, the results tend to be invariant to the particular choice of method.

10.2.2. *Creating categorical variables from continuous variables*

The categorical variables mentioned above can be derived from any source, given that they result in a unique categorization. Original Brinson-style analyses used country and sector assignments to achieve this requirement, and provide economically intuitive systematic categories.

In practice, any continuous variable may also be used as the basis for a categorization. When using Brinson analysis with continuous variables, categories are usually formed based on quartiles or quintiles of the sorted factor, whether it is beta, EPS, market capitalization, etc. Another common approach is to rank securities based on the expected returns of the Fama–French factor model, and form categories based on these expected returns (which is described in more detail in our Data section). Using such a multivariate model to create a univariate ranking is an effective way of exploring multivariate attribution within the standard Brinson framework,

and a benchmark to which we will compare novel multivariate attribution methods developed later in this chapter.

10.2.3. *Optimizing continuous variable breakpoints to maximize systematic attribution*

If the relationship between the variable under consideration and returns is non-monotonic in the classification variable or not constant within a classification level (e.g. if the effect is confined to the first vigintile of the value of the explanatory variable), the standard Brinson analysis using equal (or nearly equal) quantiles has the potential for persistently large selection effects, which is equivalent to a lack of economically interpretable performance attribution.

As there is no reason to expect sharp performance differentials at convenient groups such as quartiles, quintiles or deciles, a data-driven procedure can be a superior way to delineate stocks in terms of their factor exposures so that the delineations both correspond to realized performance and explain performance on an economically meaningful basis. The active weights of a given portfolio in these data-driven groupings will consequentially do a better job of explaining portfolio over- or under-performance without resorting to allocating a systematic component of the return to the residual.

Our objective in using a data-driven approach to create categories is to select the groupings within our continuous x vector that best explains the active returns of the portfolio. This means we wish to select groupings so that we will produce a residual effect close to zero. Given the above desired structure, we can reformulate this objective as a constrained optimization:

$$\min_{X}\{TRE^2\}\ s.t.\ X = f(order\_rank\{x\}),\ num_{groups} = j$$

The TRE^2 term is clearly nonlinear in X, so standard linear modeling is not appropriate. A brute force search over this large solution space is prohibitively costly from a computing standpoint[2]. Several features of our problem allow us to proceed, which ease the computational burden.

2 For reference, for a 1000 asset universe and j=4, the approximate number of possible solutions is close to 1 trillion (1000 * 999 * 998 * 998). Our approach is closer to 4000.

Our approach is inspired by the classification and regression tree (CART) approach first introduced by Brieman *et al*. (1984). First, at any one branch, we only allow ourselves to segment x based solely on its ordered rank. Second, the CART-based algorithm is "greedy"; at each branch in the tree, we do not need to revisit earlier nodes. Finally, we limit the number of terminal modes to four, allowing a simple comparison of the ML-based attribution to the simple quartile-based evaluation.

Limiting the number of terminal nodes to four – i.e. limiting the algorithm to just four classification groups – also makes the results easy to interpret, especially when more than one variable is used in the classification process. Because of the tendency of machine learning processes to overfit the data, limiting the number of classes ensures that we represent common systematic exposures, as opposed to a few stocks with similar idiosyncratic returns.

Simplifying the learning to these dimensions enables the attribution to be easily understood in terms of the source of systematic allocation effects, as well as a comparison of the merits of using different variables to explain portfolio returns. The process used here can be summarized as follows:

1) rank the x *variable low to high*;

2) assign k *breakpoints equal to each of the distinct values of* x;

3) for each k*, create a corresponding* X_k *dummy matrix with two columns. The first column is set to 1 if that security is below or equal to the breakpoint, and zero otherwise. The second column is defined similarly*;

4) for each X_k*, define* TRE^2;

5) select the k *breakpoint that minimizes 4)*;

6) repeat steps 1–5 within each split until the desired number of groups J *is obtained*.

The result of this process creates a dummy matrix X, which contains optimal groupings in a convenient structure. This method allows us to separate our continuous variable into a pre-specified number of groupings, which will produce an allocation effect as close as possible to the total active realized return of the portfolio. The process is easily extended to additional x variables by looping over multiple variables concurrently within steps 1–5.

For each variable, in each time period, this enables the separation of the active return of a portfolio into two components: 1) the allocation from the classification scheme under consideration and 2) the residual component of the return. For a given number of classification groups, the relative superiority of a classification scheme will be reflected in a lower variability in the residual component as measured by its standard deviation, which will naturally translate into a higher variability of the allocation component of returns.

10.3. Data description

In this study, we use the data from the US equity market covering the period December 2010 to April 2019 for the Russell 1000 index, which basically comprise the largest 1,000 stocks that are listed in the USA. The potential candidate variables consist of the factors commonly used to explain returns in academic studies – market capitalization, beta, book-to-market (B/M) and momentum. In addition to these variables, we also take into consideration the alternative measures of value – specifically earnings yield as well as a composite measure of growth. All the variables used in this study are sourced from the MSCI-Barra USE4 risk model, and represent the value of each variable that was available at that particular point in time. As such, there is no need to adjust variables (via the commonly used lag structure) for reporting lags or look-ahead bias. Using data from a single publicly available source also makes it possible to replicate the results in this chapter easily.

In addition to these factors, we also use a Fama–French composite factor to explain security returns[3]. This composite factor is created by using the actual multivariate factor returns for each of the factors. For each month, we estimate the actual factor payoffs using a four-factor multivariate regression model. These factor payoffs are then used to create the composite four-factor return for each stock in the universe – this return in effect reflects the systematic return of each stock attributable purely to its factor exposure. A diversified portfolio in a world where the data generating process was purely these four factors would have its portfolio returns completely explained by this composite factor. The purpose of this approach was to evaluate the

3 The underlying Fama–French factors are book-to-market, market capitalization, momentum and beta.

extent to which a machine learning approach could compare with a commonly used multivariate linear regression-based factor model[4].

The portfolios considered in this chapter consist of three commonly-used indices that were specifically constructed to exploit three of the well-known anomalies in equity markets – the Russell 1000 value index, the MSCI USA minimum volatility index and an equally weighted version of the Russell 1000 index. These indices capture the anomalies most commonly targeted in institutional portfolios – value, beta and market capitalization. These indices are also different in their construction process. The equally-weighted index uses a very simple rule and includes all stocks in the parent index (the Russell 1000). The Russell value index uses a multivariate screening process to eliminate stocks and uses capitalization weighting for the remaining stocks. The MSCI USA minimum volatility index uses a risk-based optimization process with constraints linked to the capitalization-weighted index. By using indices with different underlying construction processes, we are also able to evaluate the robustness of the attribution schemes considered in this chapter.

10.4. Results

We analyze each of the three indices using the same methodology. For each index, we report the percentage variability of the active returns that can be explained by using four classification levels. The base case consists of using the same number of stocks in each class – essentially forming equally weighted quartile portfolios. This is compared with the machine learning classification scheme, which is applied on both a univariate basis and a multivariate basis. In both machine learning cases, the number of classification categories is limited to four – making it easy to compare the merits of differing schemes, without having to make adjustments for the additional degrees of freedom associated with using more categories. The metric of relative success used in our analysis is the ability to explain variability in the time series of active returns – often referred to as tracking error or active risk.

The results for the different indices are presented in Tables 10.1, 10.2 and 10.3. Because the different indices have different tracking errors relative to

4 While the approach used here only uses the four factors in the typical Fama–French model, there are several commercially available attribution software packages that use a similar approach for factor-based attribution. Using more factors, for example sectors or industries, would increase the power of this approach, but for a meaningful comparison, these same variables would have to be used in the ML-based approach.

the Russell 1000 (capitalization-weighted market portfolio), we report the percentage of active risk that can be explained by the allocation effect, as opposed to the absolute level. In each table, the single variable with the greatest explanatory power is highlighted. For each of the three indices considered, the primary driver of the active returns is easily identified by the machine learning approach, by virtue of being able to explain the highest proportion of active risk, demonstrating that such an approach can be used to capture systematic portfolio tilts.

The results for the equally weighted Russell 1000 portfolio are presented in Table 10.1. Using simple quartile portfolios, referred hereto as the base, the size (market capitalization) has the greatest ability to explain the active risk of the portfolio. The Fama–French composite factor does not explain active risk better than a simple size-based evaluation. Even when using the machine learning approach, none of the variables has more power than the size factor; the ability of the single factor to explain the returns is close to 97%. For the size factor, the machine learning approach produces average breakpoints of 29, 55 and 82%, instead of equally weighted quartiles (i.e. breakpoints of 25, 50 and 75%). In addition to using just one of the variables at a time, we also consider the case when more than one variable is used in creating classification groups. When using a multivariate machine learning approach with the same variables used in the Fama–French composite factor approach, almost all the active risk (99%) in the equally weighted portfolio can be explained by just four classification groups.

	Base case	**ML case**
Beta	48.0%	59.2%
B/M	20.5%	30.3%
E/P	16.0%	35.0%
Growth	7.2%	25.9%
Momentum	34.1%	49.8%
Size	83.3%	**97.0%**
FF 4 factor	82.7%	
Multivariate		99.0%

Realized active risk or tracking error of the portfolio versus the Russell 1000 is 3.74%.

Table 10.1. *Allocation effect as a percentage of active risk for equally weighted Russell 1000*

Similar results are obtained for the Russell value index, as shown in Table 10.2. For the base case, the primary variable used in the construction of the index is able to explain the majority of the variation in active returns (85.2%). The machine learning approach is slightly better in its ability to explain active risk, with average classification breakpoints of 16.8, 35.2 and 55.1% for the B/M factor. The Fama–French composite factor with standard quartiles is significantly worse than the single-factor B/M attribution, but it is a clear second best of the individual base cases considered here. The multivariate machine learning approach does have significantly more ability to explain the returns than machine learning applied only to B/M, capturing 94.5% of the active return variability with only four classifications. The fact that the relative improvement of a multivariate approach is greater than the equally weighted portfolio is consistent with the fact that the Russell value index is created using a multi-factor screening approach.

	Base case	**ML case**
Beta	29.9%	45.8%
B/M	85.2%	**88.3%**
E/P	41.6%	56.2%
Growth	49.9%	61.3%
Momentum	37.7%	50.5%
Size	5.6%	9.6%
FF 4 factor	69.3%	
Multivariate		94.5%

Realized active risk or tracking error of the portfolio versus the Russell 1000 is 2.79%.

Table 10.2. *Allocation effect as a percentage of active risk for the Russell value index*

The results for the MSCI USA minimum variance portfolio, which is the only portfolio constructed using an optimization-based approach in concert with a risk model, are presented in Table 10.3. Despite the fact that the portfolio is created using a multi-factor risk model, a large component of returns can be explained by just a single factor. For the base case, the beta factor explains 74% of the variation, again slightly better than the Fama–French composite factor approach. The improvement by more than 10% from using machine learning with beta alone as a factor is much greater than for the other two indices considered earlier. The average percentile breakpoints for

beta of 19.4, 36.9 and 64.2% suggest optimal cutoff points which reflect the fact that there is more variation in the cross section among lower beta stocks.

	Base case	ML case
Beta	74.0%	**85.1%**
B/M	11.3%	22.8%
E/P	18.0%	30.2%
Growth	14.9%	24.9%
Momentum	20.2%	31.2%
Size	3.5%	11.3%
FF 4 factor	72.7%	
Multivariate		87.8%

Realized active risk or tracking error of the portfolio versus the Russell 1000 is 6.19%.

Table 10.3. *Allocation effect as a percentage of active risk for the MSCI USA minimum volatility index*

For all the three indices, when using the simple quartile-based classification, Fama–French quartiles are always the second best in terms of explanatory power. In the absence of the ability to consider multiple dimensions, this approach represents a good benchmark by which to evaluate the merits of an alternative attribution method. Using machine learning to identify a single- or multiple-variable classification scheme offers clear incremental advantages in terms of explanatory power.

Representative examples of the multivariate machine learning classification approach for a particular time period (August 2015) are shown in Figures 10.1–10.3. The breakpoints in the classification tree refer to the percentile breakpoints. For example, in Figure 10.1, the tree explaining value index returns is displayed. The first level of the tree is based on sorting companies into high (greater than the 26.2nd percentile of B/M) and low (less than 26.2%) B/M categories. The high-value companies are then further classified, again on B/M, into high (i.e. greater than 26.2% but less than 49.8%) and very high (greater than 49.8%). The highest value companies (i.e. greater than 49.8%) are finally split into two groups based on momentum: those below the 33.5th percentile of the momentum factor and

those above this threshold. For each classification group, we show the average return for that classification group, as well as the number of observations that fall into that group. The sum of returns across all terminal nodes is equal to the total allocation effect when using these breakpoints.

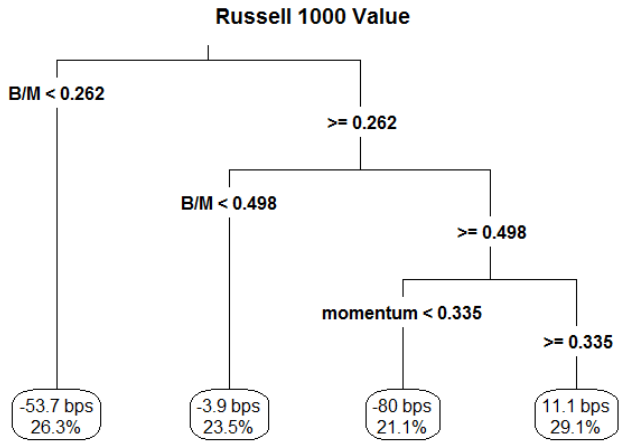


Figure 10.1. *Multivariate decomposition: August 2015*

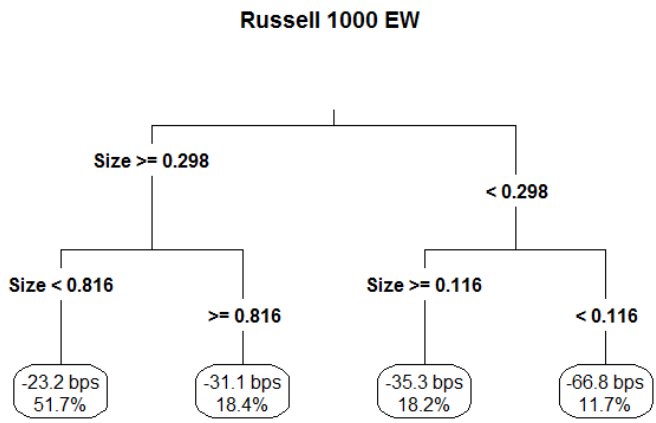


Figure 10.2. *Multivariate decomposition: August 2015*

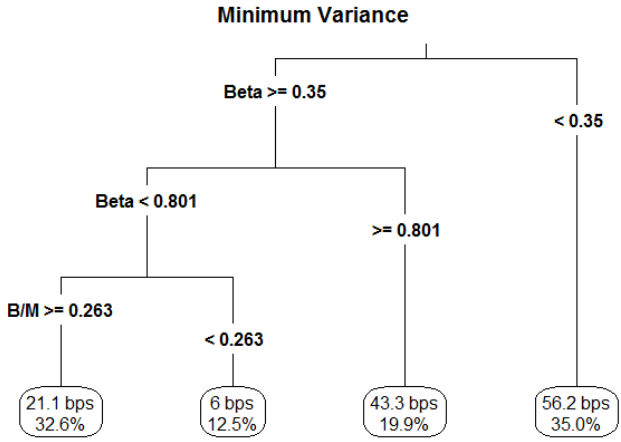


Figure 10.3. *Multivariate decomposition: August 2015*

In general, for multivariate classifications, the variable selected for the first breakpoint is the one which has the greatest power in the univariate analysis. The relative power of each factor, as it relates to each strategy, is given in Table 10.4. Variable importance here represents the decrease in residual volatility associated with using a particular variable (the most powerful variable for each index being highlighted). Beta is the second most important variable for both versions of the Russell 1000-based indices. For the MSCI USA minimum variance index, after beta, the other three variables are of relatively equal importance. This is not surprising, given that they are all used as factors in the risk model underlying the index construction.

The data presented in Table 10.4 represent the average value over the analysis period. The importance of each variable in the classification does change over time, as shown in Figures 10.4–10.6. These figures show the relative importance on a monthly basis. It is apparent that the power of the variable found to have the greatest ability to explain active returns – i.e. size for the equally weighted index, B/M for the value index and beta for the minimum volatility index – is not an artifact of a particular time period. Even the second most important variable, beta, is apparent in the graphs relating to the Russell indices.

Index	Variable	Importance
Russell 1000 equal weighted	Beta	0.185
	B/M	0.093
	Momentum	0.102
	Size	**0.620**
Russell 1000 value	Beta	0.207
	B/M	**0.528**
	Momentum	0.160
	Size	0.105
Minimum variance	Beta	**0.624**
	B/M	0.117
	Momentum	0.137
	Size	0.122

Table 10.4. *Variable importance for multivariate ML analysis*

Figure 10.4. *Variable importance by period for equally weighted Russell 1000. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

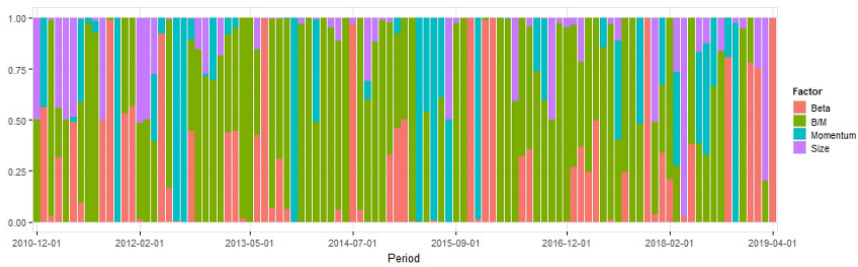


Figure 10.5. *Variable importance by period for the Russell 1000 value. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

Figure 10.6. *Variable importance by period for the MSCI USA minimum variance index. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

10.5. Conclusion

The application of machine learning-based techniques has the potential to be invaluable in separating systematic returns from idiosyncratic returns in an investment portfolio. The approach presented here represents an automated alternative to identifying and quantifying the systematic component of the active return of a portfolio. By studying three widely-used factor-based indices, we demonstrated the applicability of such an approach. By restricting the number of outcomes – here limiting the number of classification groups to just four – we were able to generate results which were intuitive and consistent with the construction of the underlying indices. Most importantly, they represent a significant improvement over commonly used attribution methods, such as the single-factor Brinson-type attribution or the regression-based multifactor attribution.

The methodology presented here has the potential to be used in the large-scale analysis of both traditional active strategies and factor-based strategies. Since the only needed inputs are portfolio and benchmark weights, security returns, and a factor library, this technique may be used in an automated manner to capture systematic portfolio tilts for a large number of portfolios. The ability of the approach to explain active return is clearly a function of the complexity of the underlying investment philosophy that determines portfolio weights – as we have the least relative success in explaining the returns of the strategy that has the highest degree of complexity (explaining 87.8% of the MSCI USA minimum volatility index) The ability to effectively explain the systematic component of the returns of traditional stock selection – which is driven by stock-specific news and not a (linear or nonlinear) factor ranking – is yet to be determined and will be the subject of future research.

10.6. References

Breiman L., Friedman J.H., Olshen R.A. and Stone C.J. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC.

Brinson Gary P. and Fachler N. (1985). Measuring Non-US Equity Portfolio Performance. *Journal of Portfolio Management*, Spring, 73–76

Carhart M.M. (1997). On Persistence in Mutual Fund Performance. *The Journal of Finance*, 52(1), 57–82.

Frongello. A.S.B. (2002). Linking Single Period Attribution Results. *Journal of Performance Measurement*, Spring, 6, (3), 10–22.

Menchero J.G. (2005). Optimized Geometric Attribution. *Financial Analysts Journal*, 61(4), 60–70.

11

Modeling Transaction Costs When Trades May Be Crowded: A Bayesian Network Using Partially Observable Orders Imbalance

Using a large database of US institutional investors' trades in the equity market, this chapter explores the effect of simultaneous executions on trading cost. We design a Bayesian network that models the inter-dependencies between investors' transaction costs, stock characteristics (bid-ask spread, turnover and volatility), meta-order attributes (side and size of the trade) and market pressure during execution, measured by the net order flow imbalance of investors' meta-orders. Unlike standard machine learning algorithms, Bayesian networks are able to account for explicit inter-dependencies between variables. They also prove to be robust to missing values, as they are able to restore their most probable value given the state of the world. Order flow imbalance being only partially observable (on a subset of trades or with a delay), we show how to design a Bayesian network to infer its distribution and how to use this information to estimate transaction costs. Our model provides better predictions than standard (OLS) models. The forecasting error is smaller and decreases with the investors' order size, as large orders are more informative on the aggregate order flow imbalance (R^2 increases out-of-sample from –0.17% to 2.39% for the smallest to the largest decile of order size). Finally, we show that the accuracy of transaction costs forecasts depends heavily on stock volatility, with a coefficient of 0.78.

Chapter written by Marie BRIÈRE, Charles-Albert LEHALLE, Tamara NEFEDOVA and Amine RABOUN.

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

11.1. Introduction

Transaction costs became of primary importance after the financial crisis. On the one hand, investment banks turned to more standardized products, switching from a high margin, inventory-driven business to a low margin, flow business, where transaction costs have to be minimized. On the other hand, the asset management industry concentrated (Haldane *et al*. 2014). A common practice is to organize the execution of large orders around one well-structured dealing desk. In 2007, the first Markets in Financial Instruments European Directive (MiFID) introduced the concept of "best execution" as a new requirement for market participants. The European best practices, including among others execution reviews, transaction costs analysis and adequate split of large orders, have spread overseas in this globalized industry.

In this chapter, we use a unique dataset of institutional investors trades: the ANcerno database, containing a large sample of asset managers' meta-orders on the US markets (Angel *et al*. 2015, Pagano 2008, Brière *et al*. 2019). While most other databases contain the meta-orders of only one asset manager, ANcerno records roughly 10% of total institutional investor activity and 8% of total daily traded volume. Because of this specificity, it is possible to estimate the "imbalance of meta-orders", i.e. the aggregated net order flow traded by investors, each day on each stock. This variable plays a role of primarily importance in the transaction costs (Capponi and Cont 2019, Bucci *et al*. 2018). Transaction costs tend to be large when you trade in the same direction as your peers, while you can even have a price improvement (i.e. obtain an average price that is lower than your decision price) if you are almost alone in front of the majority of agents trading that day. Stated differently, you pay to consume liquidity when you are part of the crowd, executing in the same direction as the market, and you are rewarded to provide liquidity to the crowd, when you are executing in the opposite direction of the market.

The specificity of this "imbalance" variable is that it cannot be observed by market participants in real time. Brokers and market makers can have a broad view of the imbalance of their clients' flows and can provide this information to the rest of the market participants with a delay, while asset management dealing desks only observe their own instructions. Therefore, the imbalance is a "*latent variable*" in the sense of Bayesian modeling. It is linked to some observable explanatory variables, and it conditions the transaction costs at the

same time. For instance: conditionally to the fact that the investor trades a buy meta-order (rather than a sell one), the imbalance is more likely to be large and positive. This dependence can be inferred using Bayes' rule.

In this chapter, we show how to use a specific model belonging to the large toolbox provided by machine learning: the Bayesian network, adapted to this kind of conditioning, to predict transaction costs, taking into account market information and trade characteristics. This class of models has been created in the golden age of machine learning (Jordan 1998); it is also known as graphical models, and has recently been used to model analyst predictions (Bew *et al*. 2018). Such models have two very interesting characteristics. First, they are able to handle missing data. Second, they can infer the distribution of latent variables given the knowledge of other ones. In our case, a model fitted on ANcerno data can be used to forecast transaction costs when the imbalance is no longer observable. In practice, our model could be fitted on data provided *ex post* by brokers[1]. Afterwards, given other explanatory variables and the observed transaction costs, a Bayesian network can infer the expected distribution of the imbalance on a given day. This is a natural feature of Bayes' rule: once the joint distribution of a set of variables is known, it is possible to obtain the expected value of any subset of other variables given the observations.

The goal of this chapter is to show how Bayesian networks can be used to model the relation between transaction costs and stock characteristics (bid-ask spread, average turnover and volatility), meta-order attributes (side and size of the trade) and market pressure (net order flow imbalance). This last variable will be considered as latent because it is only partially observable by investors (typically with a delay, or in real time but only on the investors' own trades). In practice, a possible way to implement our approach would probably be to implement a learning transfer: first, learn the graphical model on ANcerno or a similar database provided by brokers, then switch to a database in which the imbalance cannot be observed.

1 Brokers, exchanges and custodians are selling the delayed information on the flows they saw the previous day or week. This Bayesian modeling approach is perfectly suited to this kind of partial information.

We find that the daily order flow imbalance of institutional investors is a good predictor of transaction costs. Interestingly, because investors' trading tends to be crowded in one direction, and given the fund manager's knowledge of their own meta-order, he or she can infer the aggregate order flow of the market that day, to better forecast his or her trading costs. Stated differently, a fund manager could update his or her beliefs on order flow imbalance distribution of the day, after observing his or her own trading decision (side and size of his or her order). We find that his or her estimation is more accurate when his or her executed meta-order is large. Besides, we disclose evidence that a sell order is more informative on imbalance distribution than a buy order, probably because a crowded selling context is more informative about specific market conditions than a crowded buying context. We note that when an asset manager decides to sell a stock with high participation rate, he or she could expect a "rushing towards the exit door" behavior from his or her peers and assign a high probability for strong negative imbalance. Our finding confirms that the dominating variable for implementation shortfall forecast is indeed the order flow imbalance and not the order size. Moreover, the accuracy of transaction costs and market impact estimates are generally very low (Bacry *et al*. 2015). Practitioners have long suspected that the difficulty of estimating order transaction costs is due to the variance of price innovations that is hardly predictable. Thanks to our Bayesian framework, we prove that this is actually true. The Bayesian network explicitly models the dependencies between the variance of the residuals and the rest of the network nodes. We find that the dominant variable is, indeed, the price volatility with coefficient 0.78, while other node contribution to the variance is insignificant. This allows an investor to assess how confident he or she could be on each prediction given his or her meta-order and stock characteristics. Finally, we show that using partially observable order imbalance has value. The Bayesian network provides a better prediction of transaction costs after capturing the conditional dependencies between the nodes and the order flow imbalance, than when this information is not used at all (R^2 increase out-of-sample from 0.38% to 0.50%). Besides, the estimates get more accurate when the order size is large (R^2 is 2.39% for the tenth decile of order size compared to –0.17% for the first decile). These results can explain the recent concentration of institutional investors' executions on a few dealing desks. By executing the orders of a large and representative set of institutional investors, these dealing desks would have a better grasp of the aggregate order flow imbalance of the day.

This information, which is of paramount importance, could then be used either for predicting the transaction cost more accurately or for designing a better optimized execution scheme taking the aggregated market pressure into account.

The structure of this chapter is as follows: section 11.2 reviews the existing literature on transaction cost modeling and Bayesian networks. Section 11.3 presents the data. Section 11.4 provides empirical evidence of the influence of investors' trade size and order imbalance on transaction costs. Section 11.5 describes the Bayesian network method and its application to transaction cost modeling. Sections 11.6 and 11.7 present the conlcusion of this chapter.

11.2. Related literature

This chapter takes place at the crossing of two fields: the transaction costs and market impact literature, on the one hand, and Bayesian modeling, on the other hand.

11.2.1. *Transaction costs and market impact*

Market impact attracted the attention of academics following two papers: economists have been initiated to this crucial concept by Kyle's theoretical paper (1985), while researchers in quantitative finance have been largely influenced by Almgren and Chriss' (2001) empirical results. Kyle (1985) showed how a market maker should strategically ask informed traders (i.e. asset managers) for a cost to compensate for the difficulty in assessing the adverse selection he/she is exposed to in a noisy environment. This is typically what we observe empirically. Asset managers have to pay for liquidity demand while they can be rewarded for liquidity provision. Other market participants react to the aggregate offer or demand. This aggregate is exactly what we define as *the imbalance of meta-orders for a given day*. Kyle's essential result is that, given a linear market-maker pricing rule and within a Gaussian framework, the transaction costs paid by the aggregation of investors are linear in the size of the aggregated meta-order. *Kyle's lambda*, measuring the sensitivity of price impact with respect to volume flow, is a traditional measure of liquidity. This theoretical framework has been sophisticated recently, extending Kyle's game theoretical framework to continuous time, non-Gaussian behaviors, and allowing risk aversion in market makers' strategy (Cetin and Rogers 2007). It is now understood that

the informed trader optimal strategy is to try to hide meta-order in the noise, while the market maker has to slowly digest orders flow to try to extract the information it contains and ask for the corresponding price. However, the resulting market impact is not necessarily linear. Empirical studies that followed showed that in practice, market impact is more square root than linear in the size of the order (Collins and Fabozzi 1991, Bouchard *et al*. 2011 and Robert *et al*. 2012).

Almgren and Chriss' (2001) seminal paper showed how to split an order optimally to minimize execution cost, making the assumption of concave transient market impact. Bouchard *et al*. (2011) derived an optimal control scheme to mitigate this cost for large meta-orders. This literature is of primary importance because it answers the regulatory requirements around "proof of best execution" and provides a baseline framework to asset managers and investment banks to improve their best practices and metrics for execution. With the popularity of factor investing, the specific question of the implementation costs of investment strategies following an index or a systematic active strategy has been raised by regulators and market participants. Frazzini *et al*. (2012), Novy-Marx and Velikov (2015) and Brière *et al*. (2019) are attempting to answer the question of potential maximum capacity of a trading strategy, by modeling transaction costs for large order sizes and estimating the break-even capacity of factor-driven investment strategies.

11.2.2. *Bayesian networks*

Machine learning is an extension of statistical learning, born with Vapnik and Chervonenkis' seminal paper (1971). Following the universal approximation theorem for non-linear Perceptrons (a specific class of neural networks) with at least one hidden layer (Hornik *et al*. 1989), statisticians and mathematicians started investigating approximation schemes based on the minimization of a possibly nonconvex loss function, generally using a stochastic gradient descent (Amari 1993) to reach the global minimum while having good chances to escape from the local minima. Successes in Bayesian statistics, focused on coupling a prior and a posterior distribution via the concept of conjugate (Vila *et al*. 2000), opened the door to a mix of neural networks and Bayesian statistics, based on maximum likelihood estimations. Bayesian networks were born (see the seminal paper by Pearl (1986)).

Bayesian networks are convenient tools for modeling large multivariate probability models and for making inference. A Bayesian network combines observable explanatory variables with hidden latent variables in an intuitive, graphical representation.

In terms of applications, Bayesian networks have first been used for medical diagnosis, since they have been perceived as a natural extension of *expert systems*. Expert systems emerged with the first wave of artificial intelligence tools: deterministic decision trees. Adding some probabilistic properties to these trees and reshaping them into graphs is another way to see the emergence of Bayesian networks. These models have also been used with success in the troubleshooting of computed components, from printers (Skaanning *et al*. 2000) to computer networks (Lauritzen 2003). They played an important role in the automation of problem-solving for computer-related questions. Recently, they have been applied in finance. Bew *et al*. (2018) used Bayesian networks to combine analysts' recommendations to improve asset management decisions.

These models can very naturally capture the joint distribution of different variables, specified via a graphical model where nodes represent variables and arrows model the probabilistic dependencies. The very simple example of Figure 11.1 specifies that the stock bid-ask spread and its volatility both influence trading costs, while, at the same time, the stock volatility has an influence on the bid-ask spread (Laruelle and Lehalle 2018). The translation in a probabilistic language of this graph is the following. The *trading costs* TC follow a law $\mathcal{L}$ whose parameters Θ_{TC} are functions of the *bid-ask spread* ψ and of the *volatility* σ: $\text{TC} \sim \mathcal{L}(\Theta_{TC}(\psi, \sigma))$. The parameters of the law of the *bid-ask spread* are seen as a random variable, which is itself a function of the volatility: $\psi \sim \mathcal{L}(\Theta_{\psi}(\sigma))$.

More details on the mechanisms of Bayesian networks are given in section 11.5. At this stage, it is enough to say that *latent variables* can be added to the graph. An intermediate variable that is not always observable, but acts as a probabilistic intermediary (i.e. a conditioning variable) between observed variables, is enough to structure a Bayesian model. In the simple example of Figure 11.1, we can or cannot observe the bid-ask spread. When it is not observed, the Bayesian network will use its law $\mathcal{L}(\Theta_{\psi}(\sigma))$ to infer its most probable value, conditionally to the observed volatility. To do this, the model uses Bayes' conditional probability chain rule. In our analysis, we

always observe the bid-ask spread, but the net order flow imbalance of institutional investors' meta-orders is usually not known. This chapter proposes a Bayesian network to model and forecast transaction costs with a graphical model where the imbalance of institutional meta-orders is a latent variable.

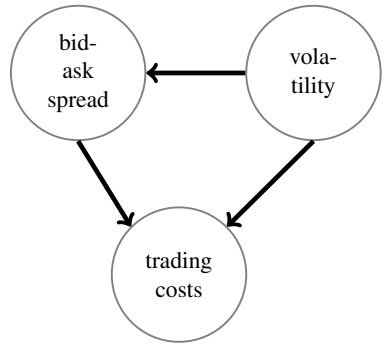


Figure 11.1. *A simple graphical model for trading costs modeling*

In summary, our chapter makes use of Bayesian networks to model the expected transaction costs of institutional investors as a function of the characteristics of the meta-order (essentially its size and direction), the market environment (stock volatility, bid-ask spread and order flow imbalance). We contribute to the current literature of trading cost estimation by proposing a methodology to account for latent variables, in our case, order flow imbalance. This variable can only be partially observed (with a delay or on a subset of all trades), but it is essential to structure the model. Our model has numerous potential applications and could be used for forecasting trading costs, estimating the capacity of a strategy or deciding on the optimal trading execution.

11.3. ANcerno database

We obtained institutional trading data for the period from January 1, 2010 to September 30, 2011 from ANcerno Ltd. ANcerno, formerly Abel Noser Corporation, is one of the leading consulting companies in providing transaction cost analysis (TCA) in the USA. It provides equity trading cost analysis for more than 500 global institutional investors, including pension

funds, insurance companies and asset managers. This database was largely used by academics to investigate institutional investors' trading behavior (see, for example, Anand *et al*. 2011, Puckett and Yan 2011 and Eisele *et al*. 2017). ANcerno clients send their equity trades in order to monitor their execution quality. ANcerno systematically reports all equity trades it receives. Therefore, costs estimated on ANcerno are representative of what is effectively paid by institutional investors. Besides, previous research showed that ANcerno is free from any survivorship or backfill bias (see Puckett and Yan 2011), which constitutes approximately 8% of the total CRSP daily dollar volume (Anand *et al*. 2013) and 10% of total institutional activity (Puckett and Yan 2011).

Hence, in our study, we use trade-level data from ANcerno on the historical composition of S&P 500 index. For each execution, ANcerno reports information on the CUSIP and ticker of the stock, the execution time at minute precision, the execution date, execution price, side (i.e. buy or sell), number of shares traded, commissions paid, whether the trade is part of a larger order, and a number of trade-level benchmarks to evaluate the quality of the execution. In our sample, we have execution data of 285 institutions (i.e. ANcerno clients). They could be either an individual mutual fund, a group of funds, or a fund manager subscribing to the Abel Noser analytical service. Each institution has one or several accounts. In our sample, we successfully track the activity of almost 44 thousands of accounts, responsible for 3.9 trillion dollars of transactions, and using the service of 680 different brokerage firms. Compared to market volume reported in CRSP, the ANcerno accounts for an average of 4.5% over the whole period. The traded amount reported in ANcerno is over a trillion dollars every year and is, therefore, large enough to be relevant. We complement the ANcerno database with daily bid-ask spread obtained from Reuters Tick History (RTH).

Consistent with machine learning best practices, we split our sample to a training set accounting for 70% of the meta-orders and a testing set accounting for the remaining 30%. The training set is chosen randomly from meta-orders in our sample, such as the number of buy orders and sell orders is equal. This procedure is very important for our study in order to estimate a non-biased net order flow imbalance. In the case of unbalance number of buy and sell orders in the training set, the prior distribution of order flow imbalance will be artificially skewed towards positive values if the number of buy orders is higher or towards negative values otherwise. The training set is used to compute the

results of sections 11.4 and 11.5, while the testing set is used for the out-of-sample predictions in section 11.6.

11.4. Transaction cost modeling

We measure trading costs with the traditional measure of implementation shortfall (Perold 1988). This is the difference between a theoretical or benchmark price and the actual traded price effectively paid for the execution, in percent of the benchmark price. In our study, we define the reference price as the last visible price before the start of the execution (arrival price). The implementation shortfall measures the total amount of slippage a strategy might experience from its theoretical returns. In essence, our cost estimate measures how much of the theoretical returns of a strategy can actually be achieved in practice.

For a parent ticket m of size $Q_k(m)$ split into N_{trades} child tickets[2] of size $v_{k,m}(i)$ executed at date d in the direction $s_k(m)$, the implementation shortfall is calculated as follows:

$$\text{IS}_k(m,d) = \frac{s_k(m)}{P_k(0)} \left(\sum_{i=1}^{N_{\text{trades}}} \frac{v_{k,m}(i)}{Q_k(m)} \times P_k(i) - P_k^{\text{ref}} \right) \qquad [11.1]$$

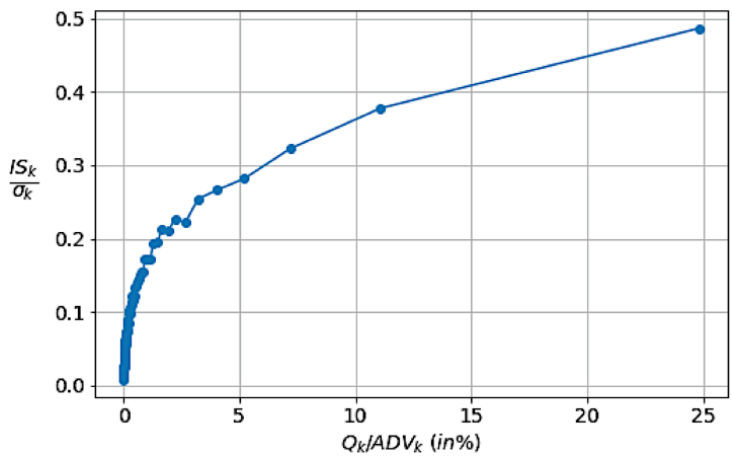
where $P_k^{\text{ref}} = P_k(0)$ is the reference price (in our case, the arrival price as provided by ANcerno). In this section, we investigate the effect of order size and order flow imbalance on the implementation shortfall of investors' transactions.

11.4.1. *Order size*

Kyle (1985) introduced the concept that trades by a market participant may have an impact on the market price. Market impact is a direct consequence of the order size effect. A large meta-order may move the price in an unfavorable direction for the trader, resulting in a higher implementation

2 Orders in ANcerno (parent tickets) are split within the execution period into smaller orders (child tickets). For each child ticket, ANcerno reports the executed volume, the price and time of execution.

shortfall. The execution cost is then increasing with order size. A series of empirical studies followed Kyle's theoretical work to confirm the existence of order size effect (Torre and Ferrari 1999, Moro *et al*. 2009, Gomes and Waelbroeck 2015, Bacry lem et al. 2015, Brière *et al*. 2019). To illustrate this effect, we regroup ANcerno tickets in 100 bins based on the participation rate Q/ADV, where ADV stands for average daily volume, and plot the average implementation shortfall scaled by price volatility of the tickets for each bin in Figure 11.2. The scaling with stock's price volatility makes estimates of implementation shortfall comparable through time and across the universe of stocks. Otherwise, we cannot compute the average on each bin as the effect of participation rate is not the same for large bid-ask spread stocks than those with small bid-ask spreads. ANcerno tickets show a concave relation between the implementation shortfall and order size relative to daily traded volume. We observe a sharp increase in the costs from 0 to 0.2 points of price volatility when order size increases from 0.01% to 2% of the average daily volume. The slope decays afterwards. For instance, a ticket with 14% participation rate costs on average 0.4 points of volatility. A power law function captures well the dependence on order trading costs to order size.



Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. We split our sample into 100 bins based on meta-order participation rate $Q_k(m)/\text{ADV}_k(d)$ and plot the average implementation shortfall scaled by stock's volatility IS_k/σ_k for each bin (blue dots).

Figure 11.2. *Order size effect on trading costs*

11.4.2. *Order flow imbalance*

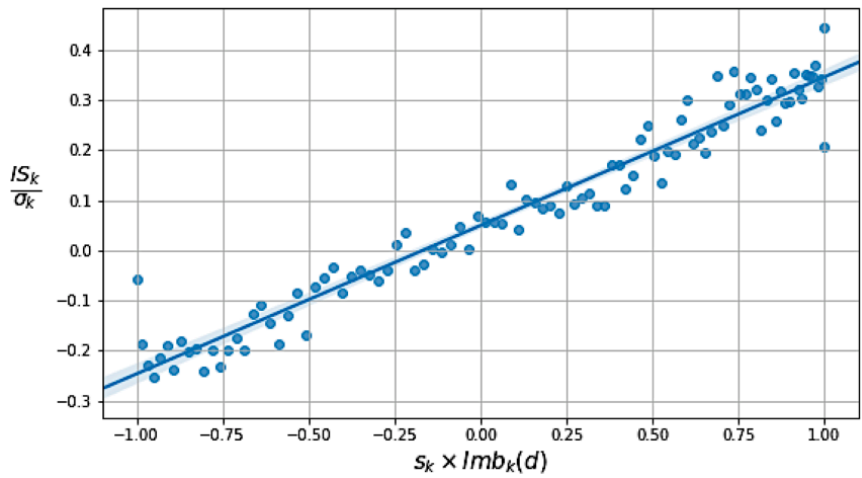
While most trading cost models emphasize the historical dependence of market impact on stock liquidity and order size, it is only recently that order flow imbalance has been recognized as a significant factor in explaining the magnitude of order transaction costs. Using ANcerno database, Capponi and Cont (2019) compared the explanatory power of order size to the effect of a proxy of market pressure "order flow imbalance" on transaction costs. They came to the conclusion that investors should focus on modeling the aggregate dynamics of market pressure during the execution period, rather than focusing on optimizing market impact at a trade-by-trade level. Moreover, market pressure is contributed by all market participants at the trading session. However, the traders who are responsible for executing institutional investors' orders contribute the most to this pressure and should be specifically taken into account in price movement forecast and transaction cost modeling. These market participants have the same profile as the informed/insider trader introduced by Kyle (1985). By the end of the trading session, the private information, which was once detained by the insider, spread to the market and became incorporated into the price level. Bucci *et al*. (2018) argued that price market impact is a function of the aggregate net volume, therefore shared indiscriminately between all market participants. Consequently, a small-sized order would cost nearly the same implementation shortfall as a much larger order if executed in the same direction during the same time frame.

We introduce the net order flow imbalance to investigate the impact of institutional investors' synchronous trading on the implementation shortfall. For a meta-order m executed at date d, the net investors' order flow imbalance is defined as the ratio of net volume executed by the other investors at day d over their total traded volume:

$$\mathrm{Imb}_k(m,d) = \frac{\sum_{m' \neq m} Q_k(m',d).s_k(m',d)}{\sum_{m' \neq m} Q_k(m',d)} \qquad [11.2]$$

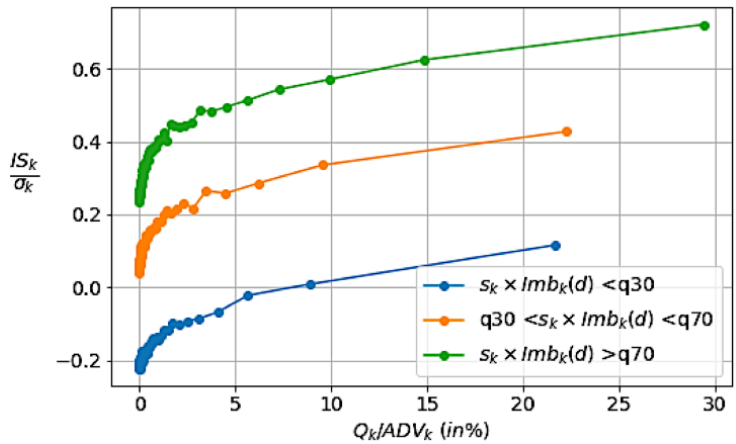
where k designs the stock, $s_k(m',d)$ is the side of the meta-order m' (i.e. 1 for buy orders and –1 for sell orders) and $Q_k(m',d)$ is its size.

Figure 11.3 shows the dependence on the implementation shortfall to institutional investors' trading imbalance. First, we note that the relationship is linear. The stronger the absolute imbalance, the higher the absolute value of price deviation during the execution. But depending on whether the trade is on the same direction as the net order flow imbalance, thus contributing to the existing market pressure, or on the opposite side, providing liquidity to the market, we could either expect to pay a significant trading cost up to 0.4 points of price volatility when investors are trading synchronously towards the same directions ($\mathrm{Imb}_k(m,d) = 1$) or benefit from a price improvement of 0.3 points of volatility when the trader is almost alone in front of his or her competitors' aggregate flow ($\mathrm{Imb}_k(m,d) = -1$). It is also worth noting that the implementation shortfall at zero imbalance is slightly positive. At neutral market pressure, the investors pay a positive transaction cost depending on stock traded and meta-order size.



Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. We split our sample into 100 bins based on net order flow imbalance multiplied by the side of the trade and plot the average implementation shortfall scaled by stock's volatility IS_k/σ_k for each bin (blue dots).

Figure 11.3. *Net order flow imbalance effect on trading costs*



Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. First, we split our sample on three buckets w.r.t. meta-order signed imbalance $(s_k(m) \cdot \mathrm{Imb}_k(m,d))$ 30% and 70% quantiles. We sort meta-orders within each bucket into 100 bins based on meta-order participation rate $(Q_k(m)/\mathrm{ADV}_k(d))$ and plot the average implementation shortfall scaled by stock's volatility IS_k/σ_k for each bin.

Figure 11.4. *Joint effect of order size and net order flow imbalance on trading costs. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

11.4.3. *Joint effect of order size and order flow imbalance*

The results in sections 11.4.1 and 11.4.2 show that the implementation shortfall depends on both the size of the executed order and market pressure during the execution period. Market pressure is approached by investors' net order flow imbalance. To disentangle the two effects (as mentioned in Figure 11.4), we split our sample on three distinct buckets with respect to meta-order signed imbalance $(s_k(m) \cdot \mathrm{Imb}_k(m,d))$ 30% and 70% quantiles. Within each bucket, we sort meta-orders into 100 bins based on meta-order participation rate $(Q_k(m)/\mathrm{ADV}_k(d))$ and compute the average implementation shortfall scaled by stock's volatility for each of the bins. Figure 11.4 plots the result, where the blue line shows the order size effect for meta-orders executed against high market pressure (signed imbalance is lower than the 30% quantile). The orange line shows the effect for meta-orders executed under standard market pressure (signed imbalance between the 30% and 70% quantiles). In contrast, the green line shows the result for orders executed in

the same direction as the market (signed imbalance larger than the 70% quantile). We observe that the impact of meta-order size is persistent in the three buckets and the power law remains valid even after conditioning on net order flow imbalance. The linear effect of the signed imbalance is visible in the difference of transaction cost level between the three buckets. This proves that these two explanatory factors do not cancel out one another. We also note that most meta-orders executed against investors net order flow benefit from a price improvement between the moment the execution starts and the moment it ends. During strongly unbalanced markets, the provider of liquidity is rewarded with a better execution price. However, for larger meta-orders ($Q_k(m)/\mathrm{ADV}_k(d) = 23\%$), the market impact of the trade prevails and the trader pays on average a positive transaction cost. The opposite is also true: when traders seek liquidity on the same direction as the remainder of the institutional investors, the trading cost gets more expensive than usual.

To further explore the joint effect of order size and net order flow imbalance on the implementation shortfall, we run the following step-wise multivariate regression. First, we perform the regression of order implementation shortfall on stock bid-ask spread and the square root of the order participation rate scale by stock volatility as described in equation [11.3]. Then, we perform a regression of the implementation shortfall on the bid-ask spread and the signed imbalance, also scaled by stock volatility (equation [11.4]). Finally, we gather the three factors on the same regression as in equation [11.5]:

$$\mathrm{IS}_k(m,d) = \alpha\ \psi_k(d) + \beta\ \sigma_k^{\mathrm{GK}}(d)\sqrt{\frac{Q_k(m)}{\mathrm{ADV}_k(d)}} + \epsilon_k(m,d) \qquad [11.3]$$

$$\mathrm{IS}_k(m,d) = \alpha\ \psi_k(d) + \gamma\ \sigma_k^{\mathrm{GK}}(d)\ s_k(m)\ \mathrm{Imb}_k(m,d) + \epsilon_k(m,d) \qquad [11.4]$$

$$\mathrm{IS}_k(m,d) = \alpha\ \psi_k(d) + \beta\ \sigma_k^{\mathrm{GK}}(d)\ \sqrt{\frac{Q_k(m)}{\mathrm{ADV}_k(d)}} + \gamma\ \sigma_k^{\mathrm{GK}}(d)\ s_k(m)\ \mathrm{Imb}_k(m,d) + \epsilon_k(m,d) \qquad [11.5]$$

where $\mathrm{IS}_k(m,d)$ is the implementation shortfall of meta-order m submitted on stock k at day d. $\psi_k(d)$ is the quoted intraday bid-ask spread of stock k

averaged on the month. $\sigma_k^{\mathrm{GK}}(d)$ is the Garman and Klass (1980) intraday volatility of stock k estimated on a 12-month rolling window. $Q_k(m)$ and $s_k(m)$ are respectively size and side (Buy/Sell) of the order. $\mathrm{ADV}_k(d)$ is the daily traded volume averaged on a 12-month rolling window, and $Q_k(m)/\mathrm{ADV}_k(d)$ is the participation rate. $\mathrm{Imb}_k(m,d)$ is the net investors' order flow imbalance estimate for order m at day d. Finally, α, β and γ are the model parameters and $\epsilon_k(m,d)$ is the respective error term.

Model	Dependent variable: $\mathrm{IS}_k(m,d)$		
$\psi_k(d)$	0.399***	0.708***	0.180***
	(0.032)	(0.028)	(0.032)
$\sigma_k^{\mathrm{GK}}(d)\ \sqrt{Q_k(m)/\mathrm{ADV}_k(d)}$	0.951***		0.712***
	(0.021)		(0.021)
$\sigma_k^{\mathrm{GK}}(d)\ s_k(m)\ \mathrm{Imb}_k(m,d)$		0.234***	0.224***
		(0.002)	(0.002)
Observations	7421548	7421548	7421548
R^2	0.005	0.016	0.017
Adjusted R^2	0.005	0.016	0.017
Residual std. error	0.017	0.017	0.017
F statistic	1892.157	5993.682	4391.073
AIC	–3964520	–3972637	–3973801
Note:	*p $<$ 0.1; **p $<$ 0.05; ***p $<$ 0.01		

Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011 on the S&P 500 historical components. $\psi_k(d)$ is the quoted intraday bid-ask spread of stock k averaged on the month, obtained from the RTH database. $\sigma_k^{\mathrm{GK}}(d)$ and $\mathrm{ADV}_k(d)$ are respectively the Garman-Klass intraday volatility and the average daily volume of stock k estimated on a 12-month rolling window. $Q_k(m)$ and $s_k(m)$ are respectively size and side (Buy/Sell) of the order. $\mathrm{Imb}_k(m,d)$ is the net order flow imbalance for order m at day d.

Table 11.1. *Transaction cost model*

The results of these three regressions are presented in Table 11.1. In the first regression, the coefficient of bid-ask spread and order size term $\left(\sigma_k^{\mathrm{GK}}(d)\ \sqrt{Q_k(m)/\mathrm{ADV}_k(d)}\right)$ are respectively 0.4 and 0.95, both statistically significant at the 1% level. Consistently with Brière *et al.* (2019)

we find that for small orders, institutional investors only pay 0.4 times the bid-ask spread. In the second regression, we replace the order size term by the market pressure term. We note that the coefficient of the bid-ask spread increases (from 0.4 to 0.7) and its confidence interval becomes tighter (lower standard deviation 0.028 vs. 0.032). The determination coefficient for the second regression is also much higher (1.6% vs. 0.5%). Finally, when we put all explanatory variables together, we find that coefficient of the order imbalance does not change (0.22–0.23) while both order size term and bid-ask spread have much lower parameters (0.18 for bid-ask spread and 0.71 for order size term) compared to the first two models. Besides, the determination coefficient of the second and third regressions is comparable. The net order flow imbalance seems to be a much better predictor of expected implementation shortfall than the size of the order. However, all coefficients are statistically significant at the 1% level.

11.5. Bayesian network modeling with net order flow imbalance as latent variable

Institutional investors' net order flow imbalance is a key factor in the estimation of meta-order transaction costs. However, this variable is only observable with a delay, for example through brokers' or custodians' reports, or on a subset of trades only (the investor's own trades). Thus, it cannot be used for production purposes. To remedy this issue, we propose a Bayesian network to incorporate all information we could get before the execution of the meta-order, and update our beliefs on the probabilistic distribution of the latent variable. We then use the most probable value of the net order flow imbalance to estimate the meta-order transaction cost. One of the interesting features of Bayesian networks is that they can be explored in both directions, thanks to Bayes' rule. Therefore, we can give an estimate of the latent variables, by probabilistic inference, before and after the variable of interest is observed. In the context of this study, this means that:

– given the characteristics of the meta-order (side and size of the trade) and stock attributes (bid-ask spread, average daily traded volume and price volatility), we can compute a first estimate of the imbalance and forecast the transaction costs that should be paid by the investor;

– once we obtain the effectively paid trading cost, we can recover a more accurate estimate of the distribution of investors' net order flow of the day,

and for example incorporate it in the estimation of order flow imbalance of the following day.

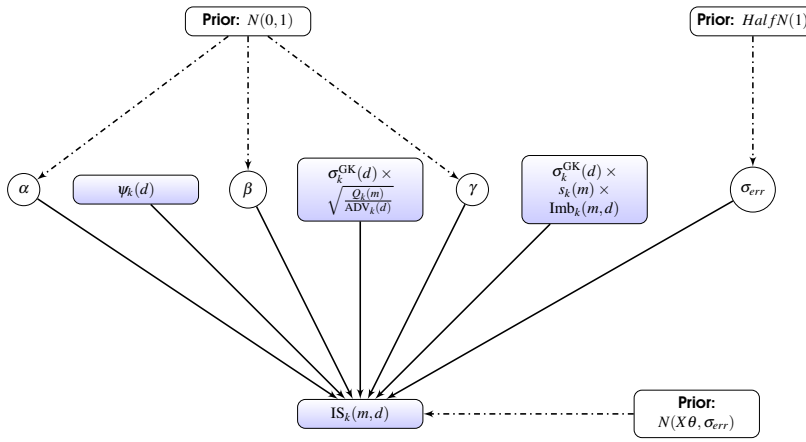
11.5.1. *Bayesian inference*

The main difference between the frenquentist approach and the Bayesian approach is that in the latter, the parameters of the models are no longer unknown constants that need to be estimated, but random variables whose parameters have to be estimated. The statistician has the possibility of incorporating his or her prior belief on the probabilistic distribution of the variable and updating his or her belief step by step as soon as new data becomes available. From one step to the other: the former "posterior distribution" is used as a prior for the next step estimate.

For instance, if y stands for the unknown random variable, x stands for the observed data, and $\mathbf{P}(y)$ stands for the prior. The posterior distribution $\mathbf{P}(y|x)$ is obtained as the multiplication of the prior $\mathbf{P}(y)$ with the likelihood $\mathbf{P}(x|y)$ of observing the data, scaled by $\mathbf{P}(x)$. The definition of conditional probabilities applied on this procedure reads:

$$\mathbf{P}(y|x) = \frac{\mathbf{P}(y)\mathbf{P}(x|y)}{\mathbf{P}(x)} \propto \mathbf{P}(y)\mathbf{P}(x|y). \quad [11.6]$$

Figure 11.5 shows how to estimate the coefficients of the Bayesian linear regression specified in equation [11.5]. First, we start by incorporating our prior beliefs, if any, on the distribution of the parameters $\theta = (\alpha, \beta, \gamma)^T$. Without any belief, a good choice is to take a non-informative prior like the normal distribution $N(0, 1)$. Hence, the best initialization for priors is a law close to the empirical repartition function of the considered variable. The variable of interest $\mathrm{IS}_k(m, d)$ follows a normal distribution centered at the estimated value $\hat{y} = X\theta$ and has variance σ^2_{err} of the error term $\epsilon_k(m, d)$. σ^2_{err} requires a non-negative prior distribution, such as the positive part of a Gaussian (i.e. *HalfNormal*) or the positive part of a Cauchy (i.e. *HalfCauchy*). The Bayesian setup gives a direct interpretation of the results: the mean of the posterior distribution is the most probable value of the parameters θ, and the 5% confidence interval is limited by the 2.5% and 97.5% quantiles of the posterior distribution.



The blue rectangle represents observed variables. The circles are the parameters that need to be calibrated. Each has a prior distribution detailed in the white rectangle.

$$X\theta = \alpha\, \psi_k(d) + \beta\, \sigma_k^{\mathrm{GK}}(d)\, \sqrt{Q_k(m)/\mathrm{ADV}_k(d)} + \gamma\, \sigma_k^{\mathrm{GK}}(d) \cdot s_k(m) \cdot \mathrm{Imb}_k(m,d)$$

Figure 11.5. *Bayesian inference of a linear regression*

MCMC (Markov Chain Monte Carlo, see (Hastings 1970) for one of the first references) methods offer an easy way to sample from the posterior, especially when the posterior does not obey a well-known expression or when we know the expression has a multiplicative term. It is very convenient for the Bayesian approach, where the posterior distribution is proportional to the multiplication of the prior and the likelihood. MCMC algorithms make computations tractable for parametric models. The intuition behind MCMC is to define a Markov chain $(x_0, x_1, \ldots)$ on the support of x, such that as when the size n of this chain goes to infinity, the new drawn point x_n is distributed accordingly to the law $\mathbf{P}_x$. The most famous algorithms to generate Markov chains having this very nice property are the Hastings-Metropolis algorithm (which is explained in section 11.13), which we use in this study, and the Gibbs sampler[3]. The marginal distribution of regression coefficients of the calibrated model is shown in the right panel of Table 11.2, while the result of the OLS regression is in the left panel. As expected from Bayesian models when the sample size is large, we end up with the same results. In addition,

3 We use the PyMC3 python package implementation of the Hastings-Metropolis algorithm described in Salvatier *et al*. (2016) with a large number of iterations $N_{iter} = 10000$.

when the priors are Gaussian, the maximum *a posteriori* of the parameters is equivalent to a ridge estimate with a quadratic regularization ($\mathbb{E}_{\theta|X,Y}[\theta] = \arg\max_\theta \mathbf{P}(\theta|X,Y) = \arg\min_\theta ||Y - X\theta||^2 + \sigma_{err}^2||\theta||^2$). This formula, similar to that of Ridge regression (see Hoerl and Kennard 1970), makes the Bayesian regression more robust to outliers than OLS. For example, it is the case for the order size term $\sigma_k^{\mathrm{GK}}(d)\sqrt{Q_k(m)/\mathrm{ADV}_k(d)}$ distribution, which explains the minor difference in coefficient estimate (0.71 vs. 0.69).

	OLS regression				Bayesian regression			
	coef	std err	Q 2.5%	Q 97.5%	coef	std err	Q 2.5%	Q 97.5%
$\psi_k(d)$	0.18	0.03	0.12	0.24	0.18	0.03	0.12	0.24
$\sigma_k^{\mathrm{GK}}(d)\sqrt{\frac{Q_k(m)}{\mathrm{ADV}_k(d)}}$	0.71	0.02	0.67	0.75	0.69	0.02	0.65	0.73
$\sigma_k^{\mathrm{GK}}(d)s_k(m)\mathrm{Imb}_k(m,d)$	0.22	0.00	0.22	0.23	0.22	0.00	0.22	0.23
RMSE (%)			1.66				1.66	
R^2 (%)			1.77				1.77	

Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011 on the S&P 500 historical components. $\psi_k(d)$ is the quoted intraday bid-ask spread of stock k averaged on the month, obtained from the RTH database. $\sigma_k^{\mathrm{GK}}(d)$ and $\mathrm{ADV}_k(d)$ are respectively the Garman-Klass intraday volatility and the average daily volume of stock k estimated on a 12-month rolling window. $Q_k(m)$ and $s_k(m)$ are respectively size and side (Buy/Sell) of the order. $\mathrm{Imb}_k(m,d)$ is the net order flow imbalance for order m at day d

Table 11.2. *OLS regression versus Bayesian regression*

11.5.2. *Bayesian network modeling*

Most of the OLS assumptions are violated. As shown in section 11.12, the marginal distribution of trading costs has a peaky shape, with fat tails (excess kurtosis of 23.46). The assumption of homoscedasticity is also violated. The variance of the error term is hardly constant across orders. Forecasting errors are smaller for small orders (implemented in a few minutes) compared to large orders (split over days) that get exposed for a longer period to market volatility. Finally, it is difficult to assume that the observations are independent of one another. Meta-orders on the same stock, regardless of the execution day, share some common variance related to the stock characteristics. Similarly, orders executed at the same trading session on different stocks face the same market conditions, and thus cannot be considered independent of one another.

In addition, Bayesian networks have the advantage of not relying on normal error distributions (Zuo and Kita 2012), as do most other machine learning algorithms. Furthermore, Bayesian networks have the advantage of giving a human-readable description of dependencies between considered variables, whereas other more complex models, such as neural networks, suffer from being considered as "black box" models.

11.5.2.1. *The structure of our Bayesian network*

Our goal is to estimate the implementation shortfall of an order. We would like to take the Bayesian network into account:

– attributes of the traded stock, such as average bid-ask spread, price volatility and average turnover;

– characteristics of the meta-order, mainly order size and side of the trade (buy/sell);

– the level of crowding during the execution: the net order flow of large institutional firms.

Figure 11.6 shows the Bayesian network we engineered. We distinguish three key dependencies. First, the bid-ask spread depends on the level of stock volatility. Second, the marginal probability distribution of order flow imbalance is a function of the meta-order size and side. Finally, the implementation shortfall is a a function of all network nodes. In the following section, we detail the nature of these dependencies and we set the priors for each group of variable separately.

11.5.2.2. *Bid-ask spread dependencies*

The relation between stock volatility and bid-ask spread is well documented. Theoretically, it is justified by Wyart *et al.* (2008) that, deriving the P&L of traders submitting market orders and those submitting limit orders, an equilibrium price is only achievable if the bid-ask spread is proportional to price volatility (i.e. $\psi_k(d) \propto \sigma_k^{\mathrm{GK}}(d)$). In the same fashion, Dayri and Rosenbaum (2015) studied the optimal tick size, and found that in the bid-ask spread the market would prefer to pay, if not constraint by the tick size verifies $\frac{\psi_k(d)}{2} \propto \frac{\sigma_k^{\mathrm{GK}}(d)}{\sqrt{M}}$. The rational is that market makers, setting the

best limits of the order book, accept to provide tight bid-ask spreads not only when the volatility (i.e. the risk of a given inventory level) is low, but also when they have more opportunities within the day to unwide their position. The relation between the bid-ask spread and the volatility is confirmed empirically on our data, as shown in Figure 11.11 of section 11.9.

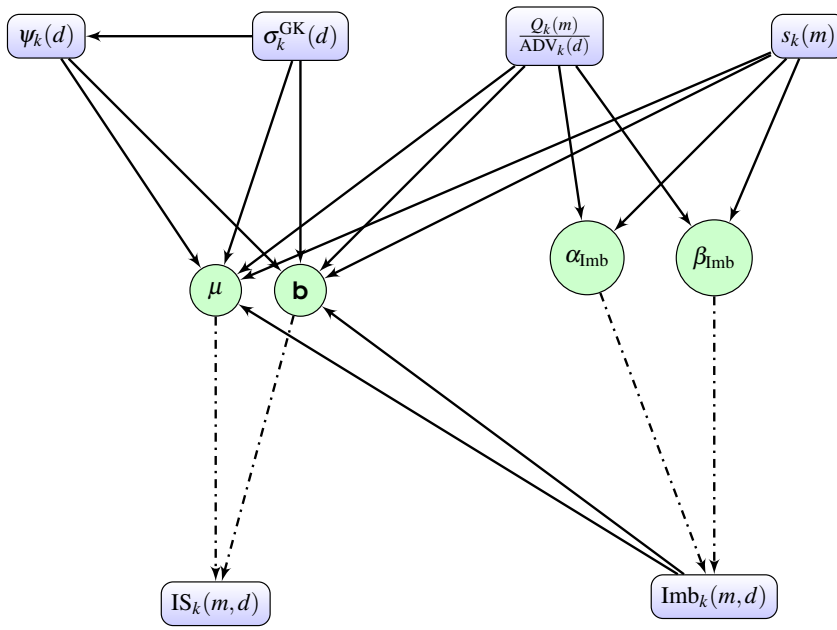


Figure 11.6. *Bayesian network for transaction costs modeling*

Consistent with the literature of stochastic models for volatility, we set the prior of stocks volatility to a log normal distribution $\sigma_k^{\mathrm{GK}}(d)) \sim \mathit{LogNormal}$. Consequently, the bid-ask spread should also follow a log normal distribution, and the conditional probability of bid-ask spread given price volatility is detailed in equation [11.7], where $c^{\psi\sigma}, \rho^{\psi\sigma}, \sigma_{\psi,\sigma}$ are the model parameters:

$$\psi_k(d)\big|\sigma_k^{\mathrm{GK}}(d) \sim \mathit{LogNormal}\left(c^{\psi\sigma} + \rho^{\psi\sigma} log(\sigma_k^{\mathrm{GK}}(d)),\ \ \sigma_{\psi,\sigma}\right) \quad [11.7]$$

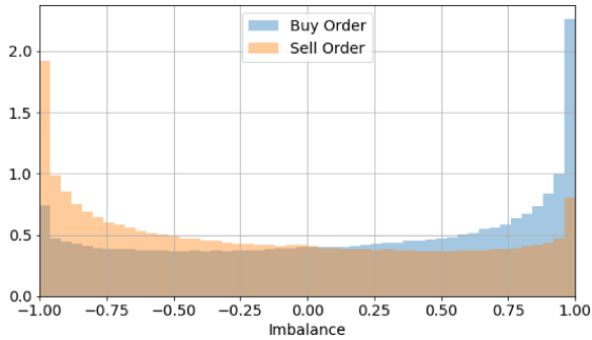
	$\mathbb{E}[X]$	Std$[X]$	Q2.5%	Q97.5%
$c_{\psi,\sigma}$	–4.137	0.006	–4.150	–4.126
$\rho_{\psi,\sigma}$	0.777	0.001	0.775	0.780
$\sigma_{\psi,\sigma}$	0.402	0.000	0.401	0.402

This table summarizes the posterior distribution of model parameters described in equation [11.7]. $\mathbb{E}[X]$, std(X), Q2.5% and Q97.5% are respectively the mean, the standard deviation, the 2.5% and 97.5% quantiles of parameter posterior distribution. The results are obtained from Hastings-Metropolis sampler with $N_{iter} = 10000$ iterations (PyMC3 implementation). The institutional investors' trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011.

Table 11.3. *Bayesian inference: bid-ask spread, volatility dependencies*

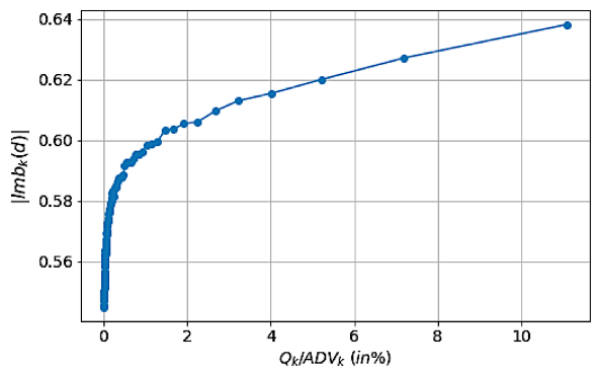
11.5.3. *Net order flow imbalance dependencies*

In this section, we quantify the dependence on net order flow imbalance to the remainder variables in the network. Figure 11.7 shows the marginal distribution of the imbalance depending on the side of the meta-order. The U-shape of the plot confirms that institutional investors have indeed correlated executions, and tend to execute the same stocks towards the same directions during the same periods, which intensify the pressure on price movements. This correlation in trade execution can be explained by various factors. Asset managers compete for the same base of customers and can implement similar strategies (Greenwood and Thesmar 2011, Koch *et al*. 2016). Thus, they face similar inflows and outflows, depending on liquidity needs and investment opportunities. Moreover, the asset management industry is subject to a series of regulatory constraints that can push funds to buy or sell simultaneously the same kind of assets. We also note that the U-shape is decomposed into two skewed distributions depending on the side of the meta-order. Therefore, given the side of the meta-order of an institutional investor, the remainder of large arbitrageurs' executions constitute either a positively (for sell) or negatively (for buy) skewed imbalance distribution. Besides, conditional on the level of a meta-order participation rate, Figure 11.8 shows that the intensity of absolute net order flow imbalance of investors meta-orders gets stronger, a confirmation of institutional investors' crowding.



Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. Given a meta-order m submitted by an institutional investor, the figure plots the distribution of the net order flow imbalance generated by the remainder of investors as defined in equation [11.2] given the side $s_k(m)$ of meta-order m.

Figure 11.7. *Net order flow imbalance distribution given meta-order side. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*



Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. We sort meta-orders into 100 bins based on the meta-order participation rate ($Q_k(m)/\mathrm{ADV}_k(d)$) and plot the average absolute net order flow imbalance for each of the bins.

Figure 11.8. *Net order flow imbalance as a function of participation rate*

The side takes two values, buy and sell. It is modeled with a Bernoulli distribution $s_k(m) \sim \textit{Bernoulli}(p_{side})$ of parameter $p_{side} = \frac{1}{2}$. The data shows that a beta function is a good approximation of the U-shape for variables defined between [0,1]. After applying the linear transformation $x \to \frac{x+1}{2}$, $\textit{Beta}(\alpha, \beta)$ is a plausible distribution of the transformed net order flow imbalance. Moreover, the beta distribution has the particularity to produce different shapes depending on the parameters α and β. It produces a symmetric U-shape when $\alpha = \beta$ and $(\alpha - 1)(\alpha - 2) > 0$, a positive skew when $\alpha < \beta$ and a negative skew when $\alpha > \beta$.

The probability density function of the transformed order flow imbalance $\mathrm{PDF}_{\mathrm{Beta}}$ is given by:

$$\mathrm{PDF}_{\mathrm{Beta}}(\alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha, \beta)} \quad \text{where} \quad \mathrm{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \qquad [11.8]$$

$$\mathbf{P}\left(\mathrm{Imb}_k(m,d) \middle| s_k(m), \frac{Q_k(m)}{\mathrm{ADV}_k(d)}\right) = \mathrm{B}(\alpha_{\mathrm{Imb}}, \beta_{\mathrm{Imb}}) \qquad [11.9]$$

The dependence on order side and order participation rate should be taken into account at the level of the parameters of the beta function (α, β).

$$\alpha_{\mathrm{Imb}} = c^{\alpha} + \rho_s^{\alpha} \cdot s_k(m) + \rho_p^{\alpha} \cdot s_k(m) \cdot \frac{Q_k(m)}{\mathrm{ADV}_k(d)} \qquad [11.10]$$

$$\beta_{\mathrm{Imb}} = c^{\beta} + \rho_s^{\beta} \cdot s_k(m) + \rho_p^{\beta} \cdot s_k(m) \cdot \frac{Q_k(m)}{\mathrm{ADV}_k(d)} \qquad [11.11]$$

The result of the Bayesian inference of the imbalance dependencies is summarized in Table 11.4. When $s_k(m) = 0$ and $Q_k(m)/\mathrm{ADV}_k(d) = 0$, the posterior distribution of net order flow imbalance is given by $\mathrm{B}(0.67, 0.68)$ which produces a U-shape. This means that when the asset manager has no signal or information on price movement, he or she can only assume the synchronization of institutional activity. This is well-illustrated by the symmetric distribution and the higher probability of the extreme values of the imbalance. But once he or she detains a signal, since the process that leads to generating this signal is independent from the execution process, he or she can use their own meta-order as an observation to update their belief on the distribution of the expected market pressure. Note also that ρ_s^{β} and ρ_p^{β} are

very low compared to ρ_s^α and ρ_p^α. This is not an issue, because what determines the strength of the skew for the beta function is the difference $(\beta - \alpha)$ (see section 11.10 for beta function properties).

	$\mathbb{E}[X]$	Std$[X]$	Q2.5%	Q97.5%
c_α	0.666	0.001	0.664	0.667
ρ_s^α	0.101	0.001	0.099	0.102
ρ_p^α	0.884	0.021	0.846	0.928
c^β	0.675	0.001	0.673	0.677
ρ_s^β	0.000	0.000	4.2e-08	0.000
ρ_p^β	0.001	0.001	1.4e-07	0.003

This table summarizes the posterior distribution of model parameters described in equations [11.10] and [11.11]. $\mathbb{E}[X]$, std(X), Q2.5% and Q97.5% are respectively the mean, the standard deviation, the 2.5% and 97.5% quantiles of parameter posterior distribution. The results are obtained from Hastings-Metropolis sampler with $N_{iter} = 10000$ iterations (PyMC3 implementation). The institutional investors' trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011.

Table 11.4. *Bayesian inference: net order flow imbalance dependencies*

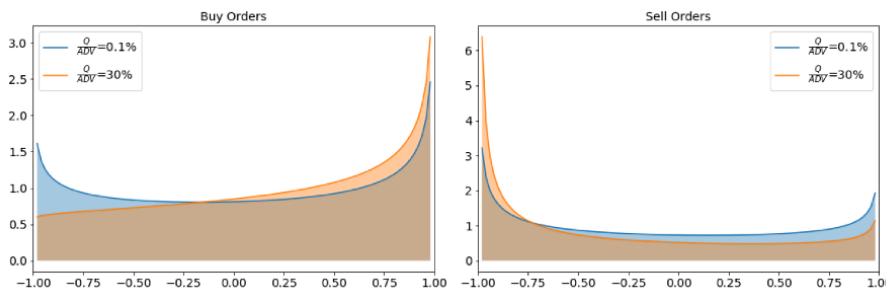
To better interpret the results of the table, we plot the posterior distribution of the net order flow imbalance, given two levels of participation rate (0.1% and 30%) for buy and sell trades. As expected, we observe that the side of the trade skews the distribution positively for a buy order and negatively for a sell order. The information of the meta-order participation rate intensifies the skew and increases the probability of having a full synchronization of investors' executions $|\mathrm{Imb}| = 1$. However, the shape of the distribution is not symmetrical between buy orders and sell orders. The skew of imbalance distribution is much stronger for sell orders. This means that when an investor is selling massively with large $\frac{Q}{\mathrm{ADV}}$, he or she could expect a high selling pressure from the market due to investors' synchronous inflows and outflows. Because institutional investors are natural buyers, implementing more long-only strategies than short selling strategies, a high selling pressure corresponds to a "rushing toward the exit door" situation. While on the opposite scenario, a buying order with high participation rate, although

informative on the market, does not give as much evidence on market participants' behavior.

11.5.4. *Implementation shortfall dependencies*

Similarly, we model the implementation shortfall as a function of all the other nodes of the network. The data shows that the historical distribution of implementation shortfall displays fat tails with pronounced non-Gaussian peaky shape. Thus, a double exponential (Laplace) probability density is a good prior of IS distribution. The probability density function (PDF) of Laplace is given by:

$$\begin{aligned} &\mathrm{IS}_k(m,d) \sim \textit{Laplace}(\mu, b), \quad \mathrm{PDF}_{\mathrm{Laplace}}(x,\mu,b) \\ &= \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right) \end{aligned} \qquad [11.12]$$



This figure shows the posterior distribution of net order flow imbalance given the meta-order characteristics. On the left panel, we plot the distribution for buy orders with two levels of participation rate, the blue line corresponds to small orders $Q_k(m)/\mathrm{ADV}_k(d) = 0.1\%$ and the orange line corresponds to large orders $Q_k(m)/\mathrm{ADV}_k(d) = 30\%$. The right panel shows the result for sell orders with the same levels of participation rate.

Figure 11.9. *Inferred net order flow imbalance given the side and the size of the meta-order. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

The location parameter μ is given by equation [11.13]. As in the linear regression, we condition the magnitude of transaction cost to stock bid-ask

spread, the participation rate scaled by the volatility and investors order flow imbalance signed by meta-order side and scaled by stock volatility. Nevertheless, we do not assume the square root function for meta-order size but a power law highlighted by the exponent γ that we estimate.

$$\mu = a_0 + a_\psi \psi_k(d) + a_{\sigma,\frac{Q}{\text{ADV}}} \exp\left\{\log(\sigma_k^{\text{GK}}(d)) + \gamma \log\left(\frac{Q_k(m)}{\text{ADV}_k(d)}\right)\right\} + a_{s,\text{Imb}}\, \sigma_k^{\text{GK}}(d)\, s_k(m)\, \text{Imb}_k(m,d) \quad [11.13]$$

Estimation accuracy is a function of market condition, speed and duration of the execution algorithm, and the aggressiveness in seeking liquidity. This heteroscedasticity of the implementation shortfall is taken into account by making the standard deviation of Laplace distribution b depend on stock attributes (spread and volatility), meta-order characteristic (participation rate) and market condition (absolute imbalance) as follows:

$$b = \exp(b_{ln}), \quad b_{ln} = b_0 + b_\psi \log(\psi_k(d)) + b_\sigma \log(\sigma_k^{\text{GK}}(d)) + b_{\frac{Q}{\text{ADV}}} \log\left(\frac{Q_k(m)}{\text{ADV}_k(d)}\right) + b_{\text{Imb}} \log\left(|\text{Imb}_k(m,d)|\right) \quad [11.14]$$

Table 11.5 summarizes the first two moments and the 2.5% and 97.5% quantiles of the posterior distribution of the parameters. First, we note that the exponent of the order size term is a bit lower than the square root $\hat{\gamma} = 0.41$, which is consistent with the previous finding of Bacry *et al.* (2015) who used a proprietary database of a broker execution in Europe. The literature usually documents a power law with exponent between 0.4 and 0.5 (Gomes and Waelbroeck 2015 and Brière *et al.* 2019). The parameters relative to the location term are consistent with the parameters estimated with OLS regression. As expected, the intercept parameter is null, the coefficient of the order size and order flow imbalance terms are similar to those estimated by the OLS regression. Only the coefficient of bid-ask spread differs significantly. The parameters of the scale of Laplace distribution are small except the stock volatility coefficient. This proves that the main contributor to the heteroscedasticity is not the order size but the stock volatility, which is

consistent with the findings of Capponi and Cont (2019) who suggested that, conditionally to the level of stock volatility and execution duration, the order size has a small impact on transaction costs.

	$\mathbb{E}[X]$	Std$[X]$	Q2.5%	Q97.5%
a_0	0.00	0.00	–0.00	0.00
a_ψ	–0.60	0.37	–1.36	0.12
$a_{\sigma,\frac{Q}{\text{ADV}}}$	0.67	0.24	0.26	1.15
γ	0.41	0.11	0.20	0.62
$a_{S,\text{Imb}}$	0.20	0.02	0.17	0.24
b_0	0.06	0.17	–0.27	0.40
b_ψ	0.09	0.03	0.05	0.14
b_σ	0.78	0.03	0.71	0.85
$b_{\frac{Q}{\text{ADV}}}$	0.05	0.01	0.04	0.06
b_{Imb}	0.01	0.01	–0.01	0.03

This table summarizes the posterior distribution of model parameters described in equations [11.13] and [11.14]. $\mathbb{E}[X]$, std(X), Q2.5% and Q97.5% are respectively the mean, the standard deviation, the 2.5% and 97.5% quantiles of parameter posterior distribution. The results are obtained from Hastings-Metropolis sampler with $N_{iter} = 10000$ iterations (PyMC3 implementation). The institutional investors' trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011.

Table 11.5. *Bayesian inference: implementation shortfall dependencies*

11.6. Forecasting implementation shortfall

We gather the different blocks of variable dependencies to constitute the Bayesian network in Figure 11.6. The parameters $(\mu, b, \alpha_{imb}, \beta_{imb})$ are estimated via Bayesian inference using the Hastings-Metropolis algorithm. Once the network is calibrated on 70% of the meta-orders, we use it to infer the latent variable of net order flow imbalance given meta-order and stock characteristics, and estimate orders implementation shortfall both in-sample (on the training set) and out-of-sample (on the testing set, the remaining 30%

of the meta-orders are not yet seen by the algorithm). Table 11.6 presents the results for both the linear regression and the Bayesian network predictions in- and out-of-sample. For the linear regression, we compare a model without order flow imbalance (equation [11.3], column 1) and one with order flow imbalance (equation [11.5], column 2). In this last model, the realized imbalance is fully observed in real time, which is never achievable in practice, but can serve as a benchmark case. We then show the results of three Bayesian networks: the first network (column 3) has never seen the information of the imbalance, neither during the training phase nor during the prediction phase. In this sense, it is comparable to the first linear regression (OLS when imbalance is not available) in the first column of Table 11.6. The second network (column 4) was trained with the information of order flow imbalance. Once this information is captured by the conditional probabilities of network edges, the network is exploited without the use of the imbalance. In this regard, order flow imbalance is partially observed. The last network (column 5) has full information on the imbalance, both at the training and testing phases, and is similar to the second OLS regression displayed in column 2. Adding information on imbalance improves the forecasting accuracy, both for the OLS regression and the Bayesian network. In-sample, it increases the R^2 from 0.52% to 1.77% for the two models, and reduces the forecasting error (RMSE from 1.67% to 1.66% and MAPE from 98.74% to 98.48%). For the same set of information (imbalance observable or not), the Bayesian network has the same accuracy as the OLS on the training set. However, the absolute average error is much smaller for the Bayesian network (–0.43 bps vs. –0.86 when imbalance is available, –1.41 bps vs. –1.43 bps when it is not). In-sample, and when all explanatory variables are observable, the Bayesian network only has a limited advantage over simple linear regressions in terms of prediction accuracy. Out-of-sample, when the imbalance is not available (panel B, columns 1 and 4), the Bayesian network is also similar to the linear regression (lower average error $= \ 0.08$ bps vs. 0.16 bps, but similar $\mathrm{RMSE} = 1.39\%$ and $\mathrm{R}^2 = 0.38\%$, and slightly higher $\mathrm{MAPE} = 99.3\%$ vs. 99.0%). But when imbalance is available (panel B, columns 2 and 5), the Bayesian network has higher forecasting accuracy than the linear regression on all criteria ($\mathrm{R}^2 = 1.20\%$ vs. 1.10%, average error $= -0.43$ bps vs. –1.08 bps, $\mathrm{RMSE} = 1.388\%$ vs. 1.389%, and $\mathrm{MAPE} = 99.41\%$ vs. 99.57%).

Imbalance	OLS regression		Bayesian network		
available	No	Yes	No	Partial	Yes
Panel A: in-sample estimation					
$\mathbb{E}[\mathrm{IS}]$ (bps)	9.020	9.020	9.020	9.020	9.020
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	7.590	8.161	7.606	8.617	8.588
$\mathbb{E}[\hat{\mathrm{IS}} - \mathrm{IS}]$ (bps)	–1.430	–0.859	–1.414	–0.403	–0.431
RMSE (%)	1.669	1.659	1.669	1.669	1.659
MAPE (%)	98.743	98.476	98.739	98.686	98.476
R^2 (%)	0.517	1.773	0.517	0.558	1.771
Panel B: out-of-sample estimation					
$\mathbb{E}[\mathrm{IS}]$ (bps)	6.394	6.394	6.394	6.394	6.394
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	6.557	5.317	6.482	8.030	5.960
$\mathbb{E}[\hat{\mathrm{IS}} - \mathrm{IS}]$ (bps)	0.163	–1.076	0.088	1.637	–0.434
RMSE (%)	1.394	1.389	1.394	1.393	1.388
MAPE (%)	99.022	99.570	99.301	99.340	99.410
R^2 (%)	0.377	1.104	0.378	0.502	1.204

Institutional trading data are obtained from ANcerno Ltd on the period ranging from January 1, 2010 to September 30, 2011. In-sample predictions are computed on 70% of the data such as the number of buy orders being equal to the number of sell orders. The remaining 30% serves for the out-of-sample prediction. RMSE and MAPE are respectively the root mean squared error and the mean absolute percentage error of the estimates.

Table 11.6. *Performance of the Bayesian network compared to the standard OLS regression*

The Bayesian network is particularly valuable when a subset of variables are only partially observable. In this case, the network captures the conditional dependencies between the nodes, and fills the missing information with the most probable values of the latent variables. In our case, the realized imbalance is not used for the prediction, but the Bayesian network is trained on imbalance to infer its distribution given meta-order characteristics. This gives a better forecast for the realized transaction cost, both in-sample and out-of-sample (e.g. higher $\mathrm{R}^2 = 0.56\%$ vs. 0.52% in-sample, 0.50% vs. 0.38% out-of-sample) than OLS or Bayesian networks that could not rely on this information.

Table 11.7 provides similar results to those in Table 11.6, but for ten deciles of order size, and for Bayesian networks using partial or full

information on imbalance. We split the training and testing sets in 10 bins with respect to the training set order size. The first bin contains small orders, lower than 0.01% of daily volume, while the last one contains very large orders, higher than 4.34% of daily volume. We assess the accuracy of the Bayesian network within the three configurations of information availability (order flow imbalance fully, partially or not available). Consistent with intuition, we find that the inferred order flow imbalance distribution is more accurate when the investor holds a larger order. The posterior distribution of order flow imbalance given a small order is a symmetric U-shape function (Figure 11.9). At best it is slightly skewed, either positively or negatively, depending on the direction of the order. Thus, the larger the investors' trade, the more informative it is on the estimation of order flow imbalance, and as a result, the more accurate is the forecast of the resulting implementation shortfall. We observe that the R^2 increases steadily, regardless of the configuration of information availability (partial or full), in-sample and out-of-sample, starting at the seventh bin. For example, the in-sample estimation of transaction costs, when imbalance is partially available, goes from an R^2 of 0.18% for the seventh decile to 2.13% for the tenth decile, while smaller deciles of order size have relatively small R^2 (from –0.03% to 0.09% for the first four bins). In fact, for small order size, the market impact is very limited and disappears in market noise. Even though the Bayesian network is trained using information on order flow imbalance, it has no advantage when the investor uses its trades attributes, if he or she only trades small order sizes. In other words, it is difficult to make good prior predictions of the order flow and thus the transaction cost when executing small orders. However, we see how information on the investors' own orders becomes more informative on the aggregate net order flow as the investors' own order size gets larger. This is in line with the recent concentration of institutional investors' executions on few dealing desks. Because the large dealing desk has a more accurate picture on investors' order flow imbalance of the day, it can assess the expected transaction cost to more accurately and potentially design a better optimized executing scheme using this information. Note also that the RMSE does not drop, because higher order size bins have few orders with a large implementation shortfall that increases the average transaction cost for the bin. This is visible in the difference between the mean and the median realized trading cost (30.64 bps vs. 24.79 bps in-sample for the tenth bin and 1.89 bps vs. 1.67 bps for the first bin). On the contrary, the MAPE, not suffering from this bias, becomes smaller as the order size increases.

Bins	1	2	3	4	5	6	7	8	9	10
$\mathbb{E}[\frac{Q}{ADV}]$ (%)	0.01	0.02	0.03	0.04	0.06	0.09	0.15	0.28	0.64	4.34

Panel A: in-sample Bayesian estimation

					Effective trading costs					
$\mathbb{E}[\mathrm{IS}]$ (bps)	1.89	2.87	2.61	4.85	6.88	7.82	7.01	9.25	16.50	30.64
$Q50$ (bps)	1.67	2.67	2.34	2.89	5.09	6.11	5.99	8.06	11.54	24.79
					Imbalance partially available					
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	4.26	4.52	4.82	5.25	5.78	6.51	7.58	9.32	12.59	25.62
RMSE (%)	1.41	1.44	1.51	1.52	1.56	1.61	1.68	1.74	1.92	2.17
MAPE (%)	100.19	99.78	100.11	99.69	98.92	98.59	98.46	97.86	96.91	93.60
R^2 (%)	–0.03	0.04	–0.00	0.09	0.17	0.24	0.18	0.27	0.77	2.13
					Imbalance available					
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	3.72	4.09	4.36	4.86	5.54	6.36	7.63	9.54	13.07	26.80
RMSE (%)	1.39	1.43	1.50	1.51	1.55	1.60	1.67	1.73	1.90	2.17
MAPE (%)	98.76	98.72	98.76	99.02	98.66	98.70	98.59	98.54	98.06	96.89
R^2 (%)	1.50	1.69	1.43	1.37	1.55	1.35	1.33	1.35	2.09	2.89

Panel B: out-of-sample Bayesian estimation

					Effective trading costs					
$\mathbb{E}[\mathrm{IS}]$ (bps)	–1.44	–0.14	0.75	2.70	3.80	4.83	8.09	8.81	12.71	28.09
$Q50$ (bps)	0.00	0.00	0.00	1.65	1.28	3.62	5.49	6.73	8.01	19.09
					Imbalance partially available					
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	4.53	4.77	5.05	5.39	5.87	6.50	7.46	8.98	11.85	22.73
RMSE (%)	1.22	1.24	1.27	1.37	1.32	1.35	1.40	1.45	1.61	1.71
MAPE (%)	101.28	101.25	101.23	100.99	101.33	98.96	98.23	97.59	97.22	93.65
R^2 (%)	–0.17	–0.09	–0.05	0.06	0.13	0.17	0.39	0.41	0.72	2.39
					Imbalance available					
$\mathbb{E}[\hat{\mathrm{IS}}]$ (bps)	2.19	2.28	2.69	2.98	3.67	4.33	5.57	6.97	9.91	22.13
RMSE (%)	1.21	1.23	1.26	1.36	1.32	1.35	1.39	1.45	1.60	1.72
MAPE (%)	99.47	99.69	99.51	99.78	99.73	99.37	99.62	99.38	99.23	97.95
R^2 (%)	1.18	1.25	1.15	0.85	0.92	1.18	0.81	0.96	1.34	2.08

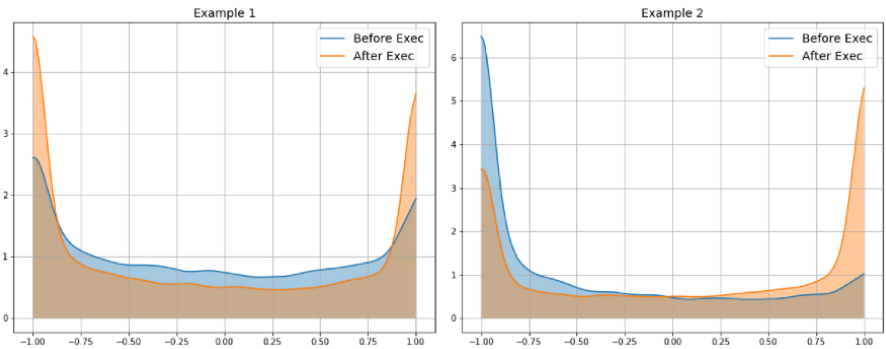
Institutional trading data are obtained from ANcerno Ltd for the period ranging from January 1, 2010 to September 30, 2011. In-sample, predictions are computed on 70% of the data such as the number of buy orders being equal to the number of sell orders. The remaining 30% serves for the out-of-sample prediction. The sample is split into 10 bins with respect to training set order size. Q50 is the 50% quantile of implementation shortfall-realized distribution. RMSE and MAPE are respectively the root mean squared error and the mean absolute percentage error of the estimates.

Table 11.7. *Performance of the Bayesian network given order size*

11.6.1. *Inference of investors' order flow imbalance given post-trade cost and market conditions*

Investors' net order imbalance is a latent variable, thus is not observable by the asset manager before executing their trade. His or her best prediction of market pressure is the inferred imbalance, after observing their own trading decision. However, their decision, although usually in line with investors' trading because of the crowd effect, can depart from what is actually traded by his or her peers. One of the interests of our Bayesian network model is that it can be used to recover the aggregate order flow imbalance prevailing during the investor's order execution, knowing his or her transaction costs. After receiving his or her transaction cost analysis report, the investor could update his or her belief on investor imbalance during their execution using the calibrated Bayesian network.

We explore two cases as an example. In the first case: the investor sells a stock $s_k(m) = -1$, with a small participation rate $Q_k(m)/\mathrm{ADV}_k(d) = 0.01\%$. His or her order is not very informative on market pressure since their trade is small, so their best estimate using the Bayesian network is a U-shape slightly skewed towards negative values of mean –0.10 (blue distribution of the left panel in Figure 11.10). Unexpectedly, the resulting trading cost is huge $\mathrm{IS}_k(m,d) = 3.02\%$ because the imbalance is very large and negative $\mathrm{Imb}_k(m,d) = -0.94$. The investor could update his or her belief on the true distribution prevailing during their execution. The posterior distribution after incorporating the realized trading cost gives higher probability to values at –1 (orange line of the left panel). The average posterior imbalance distribution is –0.17 (Table 11.8). In the second case: the investor decides to sell massively a stock, $Q_k(m)/\mathrm{ADV}_k(d) = 31.8\%$. This usually happens during market panic where other investors sell massively as well. Therefore, his or her prior distribution is highly skewed to the left ($\mathbb{E}[\mathrm{Imb}] = -0.40$, blue distribution of the right panel in Figure 11.10). While the investor expects a high transaction cost, he or she got lucky to be against the aggregate order flow ($\mathrm{Imb}_k(m,d) = 0.94$) and benefited from a price improvement of 2.6%. The posterior imbalance distribution after incorporating this information is displayed on the right panel (orange line) with a 0.05 average.



The figure plots in two panels the posterior distribution of net order flow imbalance given two examples of market conditions and order characteristics. Each time, the blue curve plots the inferred distribution when only meta-order attributes are considered, and the orange line is the updated distribution once the resulting transaction cost is observed.

Figure 11.10. *Bayesian inference of net order flow imbalance. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

	$\mathbb{E}[\hat{\text{Imb}}]$	$\text{Std}[\hat{\text{Imb}}]$	Q2.5%	Q97.5%
	Inferred imbalance: example 1			
Before exec	–0.097	0.333	–1.0	0.953
After exec	–0.175	0.381	–1.0	0.999
	Inferred imbalance: example 2			
Before exec	–0.400	0.328	–1.0	0.909
After exec	0.050	0.386	–1.0	1.000

This table summarizes the first two moments and the 2.5% and 97.5% quantiles of net order flow imbalance-inferred distribution for two scenarios before and after order implementation shortfall becomes available.

Table 11.8. *Bayesian inference of net order flow imbalance*

11.7. Conclusion

In this chapter, we use a Bayesian network to model transaction costs on US equity markets using ANcerno data, a large database of asset managers' instructions. Our main motivation is to make use of a variable of paramount importance for transaction costs, the *net order flow imbalance*. This variable

is not observed by all market participants. Brokers and market makers have access to the imbalance of their clients' flows, while dealing desks of asset managers do only observe their own instructions. Nevertheless, brokers, custodians and even exchanges started recently reselling aggregate information on their clients' flows with a delay. Bayesian networks open new perspectives to model transaction costs using *latent variables*, i.e. variables that are not always known when the model has to be used, but can be partially observed during the learning process. They make it possible to design a model that links observed and latent variables, based on conditional probabilities. The partially observable data can then be used to train the model.

Bayesian networks are able to estimate not only expected values, but also the whole probability distribution of a given variable. They are thus able to estimate the variance in the residuals of their estimation. Because of the heteroskedasticity of the error term, market impact models and transaction cost estimates traditionally have a very small R^2. A common belief among practitioners is that the effect of small mechanical price pressure is disappearing in the "market noise" (i.e. innovation on prices). We confirm this intuition in our model, by allowing the accuracy of trading cost forecasts to depend on market conditions and the investors' order characteristics. We find that the main variable which explains the variance of the residuals is the stock volatility, with a coefficient of 0.78.

Last but not least, we show several advantages of Bayesian networks for transaction cost forecasting. First, even when the latent variables (in our case, the imbalance of institutional orders at the start of the day) cannot be observed, the estimation relies on its pre-captured relationships with other observable variables (like the size and side of the investor's order to be executed). This allows the model to provide a better prediction than standard models (e.g. OLS). Second, we show that the estimates get more accurate with the size of the meta-order that the investor has to execute, because the larger the meta-order, the better the estimation of the order flow imbalance. This gives an informational advantage for large dealing desks in charge of executing the orders of numerous or large investors as they have a better picture of the aggregate imbalance. This finding is consistent with the current evolution of market practice. Small asset managers increasingly use the services of large dealing desks to benefit from this information, leading to the recent concentration of institutional investors' orders on a few dealing desks. Finally, these models can use Bayesian inference to deduce the expected

distribution of the latent variable. We show how it is feasible to ask the Bayesian network the expected distribution of large orders of other investors, either at the start or at the end of the day, once the resulting trading costs are observed.

Bayesian networks are very promising models to account for partial information. They could prove particularly valuable for "alternative datasets", such as airlines activity, web traffic or financial flows, which often provide very detailed information on a small subset of transactions. They are difficult to use in standard models, which do not accept missing values. Bayesian networks structurally model the relationship between missing and known variables. They could naturally fill this gap.

11.8. Appendix A: Garman-Klass volatility definition

Garman-Klass estimate of the volatility uses the open, high, low and close prices of the day. This estimate is robust and very close in practice to more sophisticated estimates. The formula is given by:

$$\sigma_k^{\mathrm{GK}}(d) = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}\log\left(\frac{H_{d-t}^k}{L_{d-t}^k}\right)^2 - (2\log(2)-1)\log\left(\frac{C_{d-t}^k}{O_{d-t}^k}\right)^2} \quad [11.15]$$

where the indexation k refers to the stock, d refers to the calculation day and N is the length of the rolling window in day, which in our case is 252 trading days. O_t^k, H_t^k, L_t^k and C_t^k are respectively the open, high, low and close prices of stock k at day t.

11.9. Appendix B: bid-ask spread and volatility distribution dependencies

The left panel in Figure 11.11 shows the scatter plot of the log bid-ask spread and log volatility of S&P 500 components of 2010 and 2011. It proves that the variables are related. The right panel displays the centered distributions $\left(X - \overline{X}\right)$ of log bid-ask spread and log volatility.

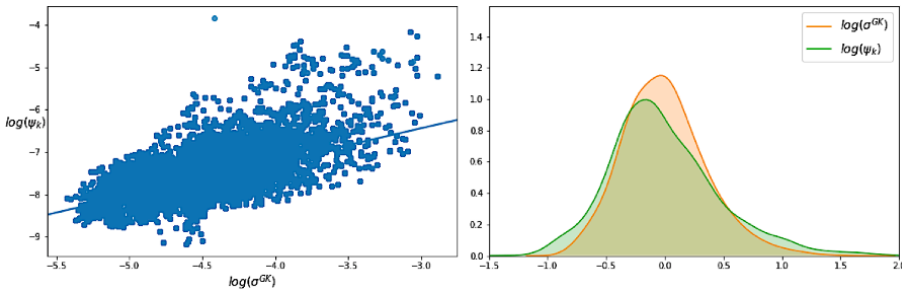


Figure 11.11. *Bid-ask spread and volatility distribution dependencies. For a color version of this figure, see www.iste.co.uk/jurczenko/machine.zip*

11.10. Appendix C: beta distribution properties

The probability density function of the beta distribution PDF_{Beta} is given by:

$$\forall x \in [0,1] \quad \mathrm{PDF}_{\mathrm{Beta}}(\alpha,\beta,x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha,\beta)} \text{ where } \quad \mathrm{B}(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \qquad [11.16]$$

The first three moments of the distribution are as follows:

$$\mathbb{E}[X] = \frac{\alpha}{\alpha+\beta} \quad \mathrm{Var}[X] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$$

$$\mathrm{Skew}[X] = \frac{2\,(\beta-\alpha)\sqrt{\alpha+\beta+1}}{(\alpha+\beta+2)\sqrt{\alpha\beta}}$$

Note that the skew of the distribution is proportional to $(\beta - \alpha)$. Therefore, when $\alpha << \beta$, the probability density function is significantly skewed towards values at 0, and in the opposite case $\alpha << \beta$, the probability density function is skewed towards values at 1. In the particular case where $\alpha = \beta$, the distribution is symmetric around the mean $\mathbb{E}[X] = \frac{1}{2}$. $(\mathrm{PDF}_{\mathrm{Beta}}(\alpha,\beta,\ \frac{1}{2}+x) = \mathrm{PDF}_{\mathrm{Beta}}(\alpha,\beta,\ \frac{1}{2}-x))$ and the skew is null. In addition, if the condition $(\alpha-1)(\alpha-2) > 0$ is fulfilled, the distribution has a U-shape. Otherwise, the beta distribution produces a concave function.

11.11. Appendix D: net order flow imbalance properties

Net order flow imbalance has a strong predictive power of daily returns. The cross-sectional average correlation for S&P 500 index components on our period of study is significantly positive up to 10.72% (Figure 11.12). Furthermore, investors' trading imbalance prevails through time. Table 11.9 shows that the daily imbalance auto-correlation decays slowly from 12.03% for the first lag to 7.44% after five days. Since order flow imbalance is only available with a delay, the long memory of the imbalance is appreciated.

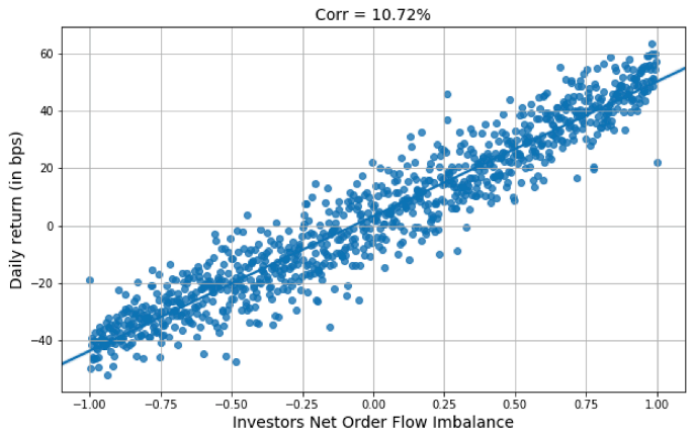


Figure 11.12. *Net order flow imbalance, daily return correlation*

	Imb_{t-1}	Imb_{t-2}	Imb_{t-3}	Imb_{t-4}	Imb_{t-5}
Imb_t	12.03	9.11	8.37	7.69	7.44

Table 11.9. *Net order flow imbalance auto-correlation*

11.12. Appendix E: implementation shortfall distribution

The implementation shortfall estimated on ANcerno meta-orders on S&P 500 components of 2010 and 2011 displays a non-normal distribution centered at 0, with standard deviation equal to 0.64, a positive skew of 0.34 and highly significant excess kurtosis of 23.46. These moments are more comparable to a double exponential distribution.

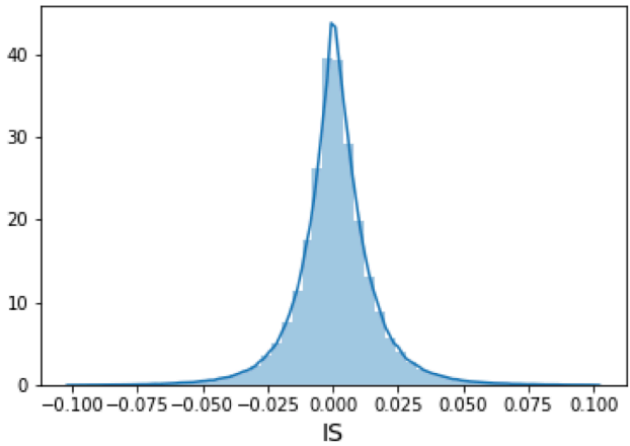


Figure 11.13. *Implementation shortfall marginal distribution*

11.13. Appendix F: Hastings-Metropolis algorithm

Hastings-Metropolis is one of the pioneer Markov Chain Monte Carlo algorithms developed in the early 1990s to sample from an unknown distribution. Given a function f proportional to the desired probability distribution $P(x)$ (a.k.a the target distribution) and a proposal distribution $q() = q(.|x)$ easy to simulate, the algorithm constructs a series of variables $(x_1, x_2, ..., x_n)$ such as given x_n:

1) generate $y_n \sim q(y|x_n)$;

2) generate $u \sim \mathcal{U}[0, 1]$ a uniform distribution;

3) compute the acceptance rate $\alpha = min\left\{\frac{f(y_n)q(x_n|y_n)}{f(x_n)q(y_n|x_n)}, 1\right\}$;

4) accept the new candidate y_n with probability α if $u \leq \alpha$. Otherwise reject.

$$X_n = \begin{cases} y_n, & \text{if } u \leq \alpha \\ x_n, & \text{otherwise} \end{cases} \qquad [11.17]$$

11.14. References

Almgren, R. and Chriss, N. (2001). Optimal execution of portfolio transactions. *Journal of Risk*, 3, 5–40.

Amari, S.-I. (1993). Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4–5), 185–196.

Anand, A., Irvine, P., Puckett, A., and Venkataraman, K. (2011). Performance of institutional trading desks: An analysis of persistence in trading costs. *The Review of Financial Studies*, 25(2), 557–598.

Anand, A., Irvine, P., Puckett, A., and Venkataraman, K. (2013). Institutional trading and stock resiliency: Evidence from the 2007–2009 financial crisis. *Journal of Financial Economics*, 108(3), 773–797.

Angel, J.J., Harris, L.E., and Spatt, C.S. (2015). Equity trading in the 21st Century: An update. *The Quarterly Journal of Finance*, 5(01), 1550002.

Bacry, E., Iuga, A., Lasnier, M., and Lehalle, C.-A. (2015). Market impacts and the life cycle of investors orders. *Market Microstructure and Liquidity*, 1(02), 1550009.

Bew, D., Harvey, C.R., Ledford, A., Radnor, S., and Sinclair, A. (2018). Modeling analysts' recommendations via Bayesian machine learning. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3269284.

Bouchard, B., Dang, N.-M., and Lehalle, C.-A. (2011). Optimal control of trading algorithms: A general impulse control approach. *SIAM Journal on Financial Mathematics*, 2(1), 404–438.

Brière, M., Lehalle, C.-A., Nefedova, T., and Raboun, A. (2019). Stock market liquidity and the trading costs of asset pricing anomalies. Research Paper No. 3380239, Université Paris-Dauphine. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3380239.

Bucci, F., Mastromatteo, I., Eisler, Z., Lillo, F., Bouchaud, J.-P., and Lehalle, C.-A. (2018). Co-impact: Crowding effects in institutional trading activity. *arXiv preprint*, arXiv:1804.09565.

Capponi, F. and Cont, R. (2019). Trade duration, volatility and market impact. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3351736.

Cetin, U. and Rogers, L. (2007). Modeling liquidity effects in discrete time. *Mathematical Finance*, 17(1), 15–29.

Collins, B.M. and Fabozzi, F.J. (1991). A methodology for measuring transaction costs. *Financial Analysts Journal*, 47(2), 27–36.

Dayri, K. and Rosenbaum, M. (2015). Large tick assets: Implicit spread and optimal tick size. *Market Microstructure and Liquidity*, 1(01), 1550003.

Eisele, A., Nefedova, T., Parise, G., and Peijnenburg, K. (2017). Trading out of sight: An analysis of cross-trading in mutual fund families. *Journal of Financial Economics*. Available at: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2023976.

Frazzini, A., Israel, R., and Moskowitz, T.J. (2012). Trading costs of asset pricing anomalies. *Fama-Miller Working Paper*, 14–05.

Garman, M.B. and Klass, M.J. (1980). On the estimation of security price volatilities from historical data. *Journal of Business*, 67–78.

Gomes, C. and Waelbroeck, H. (2015). Is market impact a measure of the information value of trades? Market response to liquidity vs. informed metaorders. *Quantitative Finance*, 15(5), 773–793.

Greenwood, R. and Thesmar, D. (2011). Stock price fragility. *Journal of Financial Economics*, 102(3), 471–490.

Haldane, G. (2014). The age of asset management? Speech, London Business School, April, 4.

Hastings, W.K. (1970). Monte Carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1), 97–109.

Hoerl, A.E. and Kennard, R.W. (1970). Ridge regression: Applications to nonorthogonal problems. *Technometrics*, 12(1), 69–82.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366.

Jordan, M.I. (1998). *Learning in Graphical Models*. Springer Science & Business Media, Germany.

Koch, A., Ruenzi, S., and Starks, L. (2016). Commonality in liquidity: A demand-side explanation. *The Review of Financial Studies*, 29(8), 1943–1974.

Kyle, A.S. (1985). Continuous auctions and insider trading. *Econometrica: Journal of the Econometric Society*, 1315–1335.

Laruelle, S. and Lehalle, C.-A. (2018). *Market Microstructure in Practice*. World Scientific, New Jersey.

Lauritzen, S.L. (2003). Some modern applications of graphical models. *Oxford Statistical Science Series*, 13–32.

Moro, E., Vicente, J., Moyano, L.G., Gerig, A., Farmer, J.D., Vaglica, G., Lillo, F., and Mantegna, R.N. (2009). Market impact and trading profile of large trading orders in stock markets. *Physical Review E*, 80, 066102.

Novy-Marx, R. and Velikov, M. (2015). A taxonomy of anomalies and their trading costs. *The Review of Financial Studies*, 29(1), 104–147.

Pagano, M.S. (2008). Which factors influence trading costs in global equity markets? *The Journal of Trading*, 4(1), 7–15.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29(3), 241–288.

Perold, A.F. (1988). The implementation shortfall: Paper versus reality. *Journal of Portfolio Management*, 14(3), 4.

Puckett, A. and Yan, X. (2011). The interim trading skills of institutional investors. *The Journal of Finance*, 66(2), 601–633.

Robert, E., Robert, F., and Jeffrey, R. (2012). Measuring and modeling execution cost and risk. *The Journal of Portfolio Management*, 38(2), 14–28.

Salvatier, J. *et al*. (2016). Pymc3: Python probabilistic programming framework. *Astrophysics Source Code Library*, 1610.016

Skaanning, C., Jensen, F.V., and Kjærulff, U. (2000). Printer troubleshooting using Bayesian networks. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, Logananthara, R., Palm, G., and Ali, M. (eds). Springer, 367–380.

Torre, N.G. and Ferrari, M.J. (1999). *The Market Impact Model*. Barra Research Insights, California.

Vapnik, V.N., and Chervonenkis, A.Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of Complexity*, Vovk, V., Papadopoulos, H., and Gammerman, A. (eds). Springer, 11–30.

Vila, J.-P., Wagner, V., and Neveu, P. (2000). Bayesian nonlinear model selection and neural networks: A conjugate prior approach. *IEEE Transactions on Neural Networks*, 11(2), 265–278.

Wyart, M., Bouchaud, J.-P., Kockelkoren, J., Potters, M., and Vettorazzo, M. (2008). Relation between bid–ask spread, impact and volatility in order-driven markets. *Quantitative Finance*, 8(1), 41–57.

Zuo, Y. and Kita, E. (2012). Stock price forecast using Bayesian network. *Expert Systems with Applications*, 39(8), 6729–6737.

List of Authors

Pere ADELL
Unigestion
Geneva
Switzerland

Daniele BIANCHI
School of Economics and Finance
Queen Mary University of London
UK

Riccardo BORGHI
Macquarie
London
UK

Kris BOUDT
Ghent University
Belgium

Marie BRIÈRE
Amundi Asset Management
and
Paris Dauphine University
France

Ryan BROWN
Wells Fargo Asset Management
Los Angeles
USA

Muzafer CELA
Vrije Universiteit Brussel
Belgium

Giuliano DE ROSSI
Macquarie
London
UK

Harindra DE SILVA
Wells Fargo Asset Management
Los Angeles
USA

Vladyslav DUBIKOVSKYY
Unigestion
Geneva
Switzerland

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Daniel GIAMOURIDIS
Scientific Implementation
Data and Innovation Group
Bank of America Securities
New York
USA

Alexei JOUROVSKI
Unigestion
Geneva
Switzerland

Emmanuel JURCZENKO
Glion Institute of Higher
Education
Switzerland

Robert KOSOWSKI
Imperial College Business School
London
UK
and
Unigestion
Geneva
Switzerland

Charles-Albert LEHALLE
Capital Fund Management (CFM)
Paris
France
and
Imperial College London
UK

Harald LOHRE
Invesco
EMP
Lancaster University
Management School
UK

Yin LUO
Wolfe Research
New York
USA

Patrick D. NEAL
Wells Fargo Asset Management
Los Angeles
USA

Tamara NEFEDOVA
Paris Dauphine University
France

Georgios V. PAPAIOANNOU
Scientific Implementation
Data and Innovation Group
Bank of America Securities
New York
USA

Sarah PERRIN
École Polytechnique
Paris
France

Amine RABOUN
Euronext Paris
and
Paris Dauphine University
France

Ravi RAMAKRISHNAN
Unigestion
Geneva
Switzerland

David E. RAPACH
Washington University
St. Louis
USA

Thierry RONCALLI
Quantitative Research
Amundi Asset Management
Paris
France

Carsten ROTHER
Invesco
University of Hamburg
Germany

Kilian Axel SCHÄFER
Metzler Asset Management
Frankfurt
Germany

Majeed SIMAAN
Stevens Institute of Technology
Hoboken
USA

Andrea TAMONI
Department of Finance
Rutgers Business School
Newark
USA

Guofu ZHOU
Washington University
St. Louis
USA

Commendations

"This volume contains some of the latest research on applications of machine-learning techniques to financial modeling, a welcome contribution to an industry that is only now beginning to fully realize the potential of modern artificial intelligence to vastly improve the way we do business and research."

Dr. Andrew Lo
MIT Sloan School of Management

"In this book, Professor Jurczenko has assembled an impressive collection of articles, covering a wide range of topics of importance to all asset managers. Professionals unfamiliar with these techniques are at a disadvantage, because the complexity of financial datasets is beyond the grasp of econometric models. The reader will gain a glimpse into the future, as machine learning continues to make progress in replacing or complementing legacy methods."

Dr. Marcos Lopez de Prado
Cornell University
and
True Positive Technologies LP

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

Index

A, B, C

Machine Learning for Asset Management: New Developments and Financial Applications, First Edition. Edited by Emmanuel Jurczenko.

D, E, F

G, H, I

L, M, N

O, P, Q

R, S, T, X

Other titles from

in

Mathematics and Statistics

2020

MANOU-ABI Solym Mawaki, DABO-NIANG Sophie, SALONE Jean-Jacques
Mathematical Modeling of Random and Deterministic Phenomena

2019

BANNA Oksana, MISHURA Yuliya, RALCHENKO Kostiantyn, SHKLYAR Sergiy
Fractional Brownian Motion: Approximations and Projections

GANA Kamel, BROC Guillaume
Structural Equation Modeling with lavaan

KUKUSH Alexander
Gaussian Measures in Hilbert Space: Construction and Properties

LUZ Maksym, MOKLYACHUK Mikhail
Estimation of Stochastic Processes with Stationary Increments and Cointegrated Sequences

MICHELITSCH Thomas, PÉREZ RIASCOS Alejandro, COLLET Bernard, NOWAKOWSKI Andrzej, NICOLLEAU Franck
Fractional Dynamics on Networks and Lattices

VOTSI Irene, LIMNIOS Nikolaos, PAPADIMITRIOU Eleftheria, TSAKLIDIS George
Earthquake Statistical Analysis through Multi-state Modeling
(Statistical Methods for Earthquakes Set – Volume 2)

2018

AZAÏS Romain, BOUGUET Florian
Statistical Inference for Piecewise-deterministic Markov Processes

IBRAHIMI Mohammed
Mergers & Acquisitions: Theory, Strategy, Finance

PARROCHIA Daniel
Mathematics and Philosophy

2017

CARONI Chysseis
First Hitting Time Regression Models: Lifetime Data Analysis Based on Underlying Stochastic Processes
(Mathematical Models and Methods in Reliability Set – Volume 4)

CELANT Giorgio, BRONIATOWSKI Michel
Interpolation and Extrapolation Optimal Designs 2: Finite Dimensional General Models

CONSOLE Rodolfo, MURRU Maura, FALCONE Giuseppe
Earthquake Occurrence: Short- and Long-term Models and their Validation
(Statistical Methods for Earthquakes Set – Volume 1)

D'AMICO Guglielmo, DI BIASE Giuseppe, JANSSEN Jacques, MANCA Raimondo
Semi-Markov Migration Models for Credit Risk
(Stochastic Models for Insurance Set – Volume 1)

GONZÁLEZ VELASCO Miguel, del PUERTO GARCÍA Inés, YANEV George P.
Controlled Branching Processes
(Branching Processes, Branching Random Walks and Branching Particle Fields Set – Volume 2)

HARLAMOV Boris
Stochastic Analysis of Risk and Management
(Stochastic Models in Survival Analysis and Reliability Set – Volume 2)

KERSTING Götz, VATUTIN Vladimir
Discrete Time Branching Processes in Random Environment
(Branching Processes, Branching Random Walks and Branching Particle Fields Set – Volume 1)

MISHURA YULIYA, SHEVCHENKO Georgiy
Theory and Statistical Applications of Stochastic Processes

NIKULIN Mikhail, CHIMITOVA Ekaterina
Chi-squared Goodness-of-fit Tests for Censored Data
(Stochastic Models in Survival Analysis and Reliability Set – Volume 3)

SIMON Jacques
Banach, Fréchet, Hilbert and Neumann Spaces
(Analysis for PDEs Set – Volume 1)

2016

CELANT Giorgio, BRONIATOWSKI Michel
Interpolation and Extrapolation Optimal Designs 1: Polynomial Regression and Approximation Theory

CHIASSERINI Carla Fabiana, GRIBAUDO Marco, MANINI Daniele
Analytical Modeling of Wireless Communication Systems
(Stochastic Models in Computer Science and Telecommunication Networks Set – Volume 1)

GOUDON Thierry
Mathematics for Modeling and Scientific Computing

KAHLE Waltraud, MERCIER Sophie, PAROISSIN Christian
Degradation Processes in Reliability
(Mathematial Models and Methods in Reliability Set – Volume 3)

KERN Michel
Numerical Methods for Inverse Problems

RYKOV Vladimir
Reliability of Engineering Systems and Technological Risks
(Stochastic Models in Survival Analysis and Reliability Set – Volume 1)

2015

DE SAPORTA Benoîte, DUFOUR François, ZHANG Huilong
Numerical Methods for Simulation and Optimization of Piecewise Deterministic Markov Processes

DEVOLDER Pierre, JANSSEN Jacques, MANCA Raimondo
Basic Stochastic Processes

LE GAT Yves
Recurrent Event Modeling Based on the Yule Process
(Mathematical Models and Methods in Reliability Set – Volume 2)

2014

COOKE Roger M., NIEBOER Daan, MISIEWICZ Jolanta
Fat-tailed Distributions: Data, Diagnostics and Dependence
(Mathematical Models and Methods in Reliability Set – Volume 1)

MACKEVIČIUS Vigirdas
Integral and Measure: From Rather Simple to Rather Complex

PASCHOS Vangelis Th
Combinatorial Optimization – 3-volume series – 2nd edition
Concepts of Combinatorial Optimization / Concepts and Fundamentals – volume 1
Paradigms of Combinatorial Optimization – volume 2
Applications of Combinatorial Optimization – volume 3

2013

COUALLIER Vincent, GERVILLE-RÉACHE Léo, HUBER Catherine, LIMNIOS Nikolaos, MESBAH Mounir
Statistical Models and Methods for Reliability and Survival Analysis

JANSSEN Jacques, MANCA Oronzio, MANCA Raimondo
Applied Diffusion Processes from Engineering to Finance

SERICOLA Bruno
Markov Chains: Theory, Algorithms and Applications

2012

BOSQ Denis
Mathematical Statistics and Stochastic Processes

CHRISTENSEN Karl Bang, KREINER Svend, MESBAH Mounir
Rasch Models in Health

DEVOLDER Pierre, JANSSEN Jacques, MANCA Raimondo
Stochastic Methods for Pension Funds

2011

MACKEVIČIUS Vigirdas
Introduction to Stochastic Analysis: Integrals and Differential Equations

MAHJOUB Ridha

Recent Progress in Combinatorial Optimization – ISCO2010

RAYNAUD Hervé, ARROW Kenneth
Managerial Logic

2010

BAGDONAVIČIUS Vilijandas, KRUOPIS Julius, NIKULIN Mikhail
Nonparametric Tests for Censored Data

BAGDONAVIČIUS Vilijandas, KRUOPIS Julius, NIKULIN Mikhail
Nonparametric Tests for Complete Data

IOSIFESCU Marius *et al.*
Introduction to Stochastic Models

VASSILIOU PCG
Discrete-time Asset Pricing Models in Applied Stochastic Finance

2008

ANISIMOV Vladimir
Switching Processes in Queuing Models

FICHE Georges, HÉBUTERNE Gérard
Mathematics for Engineers

HUBER Catherine, LIMNIOS Nikolaos *et al.*
Mathematical Methods in Survival Analysis, Reliability and Quality of Life

JANSSEN Jacques, MANCA Raimondo, VOLPE Ernesto
Mathematical Finance

2007

HARLAMOV Boris
Continuous Semi-Markov Processes

2006

CLERC Maurice
Particle Swarm Optimization