Natural Language Processing

Contents

Introduction

Natural language processing focuses on the interaction between human language and computers. It enables machines to get closer to a human level understanding of the language text. It is easy for humans to understand the text and interpret the meaning of the same. But doing the same on the large corpus of data is not feasible.

In finance and trading, every day, a large amount of data is generated, and it is challenging for humans to find useful information and make trading decisions. This data comes in the form of News, scheduled economic releases, employment figures, interest rates, inflation and GDP figures. All such data moves the market. To deal with such a large amount of data and get important information natural language processing techniques are used.

Sentiment Score

A Sentiment Score is a quantitative value assigned to a piece of content/text which is used to make trading decisions. For example, the tweet "$AAPL is my best investment so far" has a positive sentiment score of 0.6369 indicating a bullish trend for Apple stock. On the other hand, the tweet "AAPL is not my best investment so far" has the calculated score of -0.5216 which is a negative score indicating the stock fall.

Application in Sentiment Trading

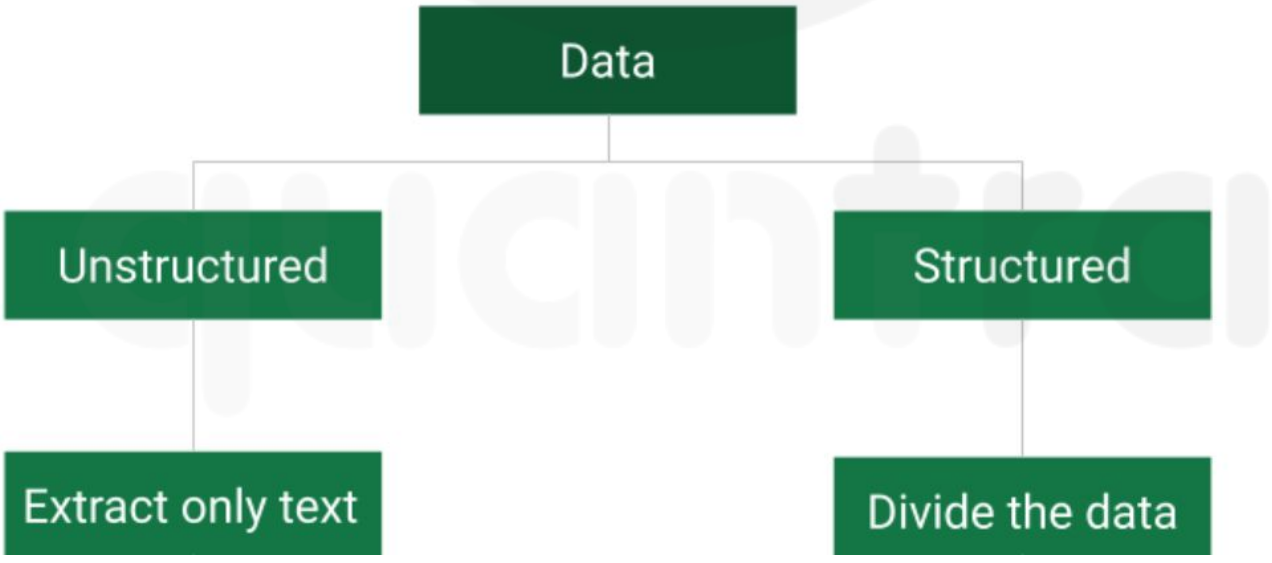
Following are the steps that one needs to follow for using NLP in sentiment trading:

- Get the data
- Preprocess the data
- Convert the text to a sentiment score
- Generate a trading model
- Backtest the model

Get the data

To build an NLP model for trading, you need to have a reliable source of data. There are multiple vendors for this purpose.

For example, Twitter and Webhose provide it for free, while others such as News API, Reuters and Bloomberg will charge you for it. Let us divide the data into two types and try to approach each of them differently.

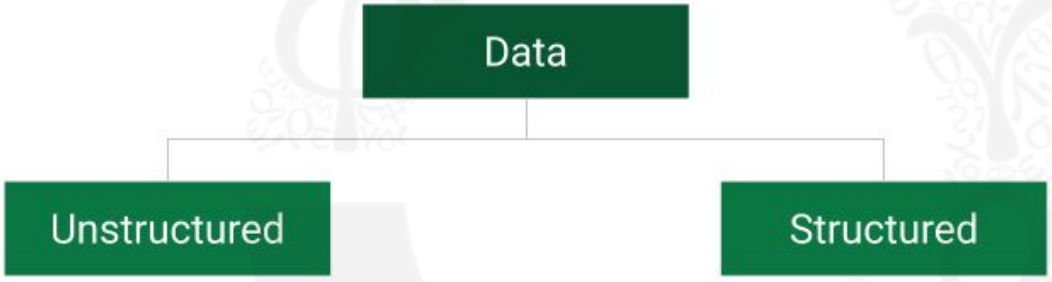


Structured data is one that is published in a predetermined or consistent format. The language is also ve consistent.

For example, the press release of fed minutes or a company's earnings can be considered as structured data. Here the length of the text is usually very huge.

On the contrary, unstructured data is one where neither the language or format is consistent. For example, twitter feed, blogs and articles can be counted as a part of this. These texts are usually limited in size.

Preprocess the data



There are different problems associated with these two data sets. Unstructured data like Twitter feeds consists of many non-textual data, such as hashtags and mentions. These need to be removed before measuring the text's sentiment.

For structured data, the size of the text can easily cloud its essence. To solve this, you need to break the text down to individual sentences or apply techniques such as tf-idf to estimate the importance of words.

Convert the text to a sentiment score

To convert the text data to a numerical score is a challenging task. For unstructured text, you can use pre-existing packages such as VADER to estimate the sentiment of the news. If the text is a blog or an article then you can try breaking it down for VADER to make sense of it.

For structured text, you don't have any pre-existing libraries that can help you convert the text to a positive or a negative score. So, you will have to create a library of your own.

When building such a library of relevant structured data, care should be taken to consider texts from similar sources and the corresponding market reactions to this text data.

For example, if the Fed releases a statement saying that “the inflation expectations are firmly anchored and changes it to “the inflation expectations are stable”, then libraries like VADER won't be able to tell the difference apart, but the market will react significantly.

To understand score the sentiment of such text you need to develop a word-to vector model or a decision tree model using the tf-idf array.

Generate a trading model

Once you have the sentiment scores of the text, then combine this with some kind of technical indicators to filter the noise and generate the buy and sell signals.

To generate these signals, you can either do it manually from your experience or use a decision tree type model.

Backtest the model

Once the model is ready, you need to backtest it on the past data to check whether your model’s performance is within the risk limitations. While backtesting, make sure that you don't use the same data that is used to train the decision tree model.

If the model confirms to your risk management criterion then you can deploy the model in live trading.