Brendon Smith

Algothon 2018 @ BlackRock by Imperial College London

MD AlphaGen

Send him a ping

**Systematic active equity** – 30 year track record of innovation

One of the pioneers of quant investing

Systematic approach to investment management with an emphasis om aiming produce consistent and reliable results

Research driven process blends investment insight with cutting edge technology

Designed to deliver differentiated and diversifying potential returns for clients

George poyiadjis – Systematic Income Alpha

**NLP Strategy by Andrea de Marco**

We are given 10 years’ worth of Reuters articles from 2007 to 2017. This amounts to approximately 2.5 million articles. In an ideal scenario, we must iterate through every single article, access the URL, generate a transcript of the actual article, preprocess the transcript, and run sentiment analysis and topic modelling in order to generate the relevance of the article to our problem and score to quantify how important the article could contribute to the inflation and deflation of the market.

Within a hackathon environment, implementation would have presented the team with several issues, including subpar performance and code efficiency, due to the rush of hacking. More importantly, accessing all those articles, preprocessing and generating a score would have taken a considerable amount of time and computational power. The code was compiled on a 1.4GHz i5 dual core MacBook Air, which is considerably old machine and hence efficiency had to be taken into consideration.

The Reuters data was stored in pickled data files containing a timestamp, title and URL for all the articles that were published per day. We choose 1546 days in total from 1 January 2014 to 31 December 2017, this is because the teams were given foreign exchange data of the mystery stock for that date range.

In order to save on memory and computational performance, the first step was to iterate through each title and put in place a filter that removes all the titles that are not relevant to the financial world. For this we used a csv file from the Loughran-McDonald Master Dictionary (https://sraf.nd.edu/textual-analysis/resources/ - identical to using pysentiment) to represent out financial dictionary. Additionally, we have also added our own custom list to this dictionary with adds important terms that were not included in the dictionary.

Furthermore, the titles went through two rounds of preprocessing where all the numbers and characters joined to these numbers were removed, all the punctuation was removed and letters were made lower case. This amounted to a corpus of titles of that was used to create a sentiment score. The next step was to generate a sentiment score that ranges from 1: positive to –1: negative based on the words included in the titles. To do this we used the package textblob. For each day, the mean from the score of each title was calculated.

Due to inefficient time, we opted to give a sentiment directly from the title. Ideally, after the titles were filtered, web scrapping could have been initialized which generates a transcript of the article. After preprocessing, the corpus would have given us a more reasonable sentiment score for each article which could then apply to our final score of each day. At this point, I should say that we were working under the assumption that a positive sentiment score would indicate stocks to rise whilst a negative sentiment score would indicated stocks to fall.

Instead, we decided to focus our efforts on generating a sentiment score based on the titles of the article per hour instead of per day. This required a bit of time to hack and iron out the bugs, but in the end we ended up with a csv file that has two columns, the timestamp per hour from January 1, 2014 and December 31, 2017; and the sentiment score from –1 to 1.

The csv file generated includes 13 days of sentiment score (there was an error on Day 14 due to Chinese letters) and took approximately 5 minutes. The whole 1546 days would have taken approximately 2 hours to compile and render the csv file.

Other ideas: Get an indication of the type of stock the mystery ticker, by looking at the prices given to us. This might be helpful in order to introduce an additional filter that analyses articles relevant to these types of stocks/tickers.

Challenges faced:

* Learning NLP as we go – this took a considerable amount of time but I felt as though this was a necessity to get a brief idea of what we were actually trying to do.
* Minor bugs had to be fixed along the way, but I feel like this would be an issue for everyone.
* Chinese titles – my guess would be to either remove characters not related to the English language (did not have time to work on this in the end)