Sentiment Analysis of News Headlines to Determine Stock Movements

Nicholas DeBaise
Union College
debaisen@union.edu

Abstract

It has been proven in studies that there is a relationship between the sentiment of news articles in the news and stock movements pertaining to the news articles. Our team set out to find out if it is possible to train a machine learning model to learn how stocks move based on the sentiment of a given news article. We used a research paper[1] as a guideline to collecting our data, developing our model, and comparing our results. We sought to see if it's possible to predict stock movements solely based on news articles for a given day. Our code is made public via Github[2].

1 Introduction

The stock market is highly volatile due to numerous amounts of factors that all contribute to stock movements. Investors, in a way, attempt to predict stock movements in order to maximize gains and minimize losses. An investor's goal is ultimately to make money from the stock market by buying low and selling high. It would be beneficial to them to predict if a stock's price will increase or decrease on a given day or over a period of time. However, predicting a stock price is an incredibly complicated and potentially inaccurate process. Due to so many factors that stocks rely on, it is difficult to model.

Yet, many studies have shown that news on a given stock can have a direct effect on the movement of that stock. Our goal is to create a machine learning model that is able to model how news sentiment affects stock movements. Specifically, we will train linear models on sentiment of news headlines and if a given stock price rose or fell on the day that the headline was published.

We will do this by following the research paper by Kalyani, Bharathi, and Jyothi: Stock Trend Prediction Using News Sentiment Analysis[1]. They achieved high accuracy of roughly 80-90%. One way we are looking to improve results is to improve the sentiment detection of headlines. In the paper, they approached this topic rather crudely. They first tokenize and remove stop-words from the given article. Next, they place all of the tokens into a bag-of-words. Finally, they have a dictionary of positive and negative words. They count up the number of positive words and subtract the number of negative words from the bag-of-words. The resulting number is their sentiment assigned to the news article.

Our goal is to use FinBERT, a fine-tuned BERT model that is trained to detect sentiment of financial news. We believe that this model will be more accurate than the crude approach taken in the paper, which will hopefully lead to better results.

2 FinBERT

While BERT for sentiment analysis is an appropriate tool for most tasks, sentiment of financial news is a complicated task due to the domain-specific language. For example, take the sentence "Stocks rallied and the British pound gained." This sentence contains niche language choices such as "rallied" and "gained" that apply to niche topics, such as the "British pound". This makes sentiment analysis a difficult topic. Dogu Tan Araci introduced FinBERT[3] to combat these issues. With FinBERT, a BERT language model is further trained on a domain-specific, pretagged corpus for sentiment of financial text.

3 Sentiment Analysis

Our program begins by scraping necessary news headlines pertaining to a given stock. It does this by accessing a data set[4] of 4 million news headlines

[1] https://arxiv.org/pdf/1607.01958.pdf
[2] https://github.com/nickdebaise/Sentiment-Analysis-CSC483
[3] https://arxiv.org/pdf/1908.10063.pdf
[4] https://www.kaggle.com/datasets/miguelaenlle/

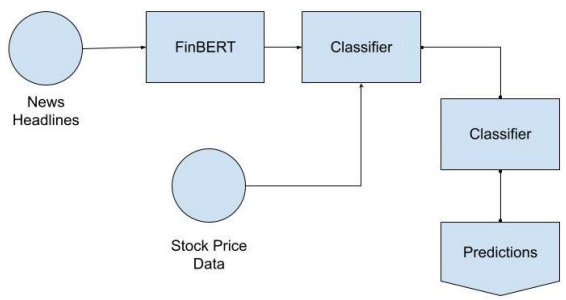

Figure 1: Diagram of our program/model

pertaining to over 6000 stocks. In particular, we use a further processed data set containing 1,407,329 entries. Once retrieving the relevant headlines, our program feeds those headlines into the FinBERT model, which gets sentiment scores for each headline. Our program then fetches the open and close price for the date that the headline was published. It classifies each open/close price as 1 or 0 based on how well the stock performed that day. *If the stock rose that day, it was assigned a class of 1. If it fell that day, it was assigned a class of 0.* Finally, it feeds this data into our classifier. The classifier is then able to predict stock price movements given a headline's sentiment scores. This process is modeled in Figure 1.

3.1 Data Processing

The data set from Kaggle contains approximately 1.4 million entries. Each entry contains a news headline, a URL to the news article, the publisher of the article, the date the article was published, and a stock's ticker that pertains to the news headline. All of this information is useful. Specifically, the ticker, news headline, and date are most important to our task. In order to maximize our training and testing data, we found that the company Merck (MRK) had the largest number of data points, containing 3,333 headlines. These headlines are then fed into FinBERT and 3 scores are received back. Each headline contains a positive sentiment score (0-1), a negative sentiment score (0-1) and a neutral sentiment score (0-1). These 3 scores already contain more information than the crude approach the research paper took which only contained a positive or negative number pertaining to each article. Finally, the dates of each headline are used to fetch the opening and closing price of the given stock on the date. All of this data is fed into a linear model such as Naive Bayes, SVM, and RandomForestClassifier.

`massive-stock-news-analysis-db-for-nlpbacktests`

3.2 Results

Different stocks yield different results – different classifiers also yield different results. With that being said, our classifiers did not perform as well as we expected and did not compare in any way to the results that Kalyani found. The following results (shown in table 1) were trained on 2333 headlines (70% of total data) and tested on 667 headlines (20% of total data) of the Merck (MRK) stock. Clearly, accuracy scores of 51% are not ideal (and hardly better than randomly guessing). They are also nowhere near what Kalyani's paper was able to achieve. Maybe MRK, a COVID-boom stock, is an anomaly. Microsoft (MS) contains the second most data and is more of a traditional, well-known and highly traded stock. The results (shown in table 2) are not much better. While accuracy increased by roughly 1%, no significant statistical difference was measured. We tried one more stock, Google (GOOGL) to see if results would change with slightly less data. Table 3 highlights the results from Google, which are slightly better, achieving a maximum accuracy of 56.6%. Clearly, something in our model is not working as intended. One thought is that we are forcing our classifier to make a decision of whether the stock is a buy or sell (class of 1 or 0). However, there are times when the stock price may not really move that day, particularly when the neutral score on a headline for that day is high. This might mean that the stock does not move much, and a separate class is necessary to predict that. We introduce a way to classify a stock as 2 (buy), 1 (hold) or 0 (sell) based on if the close price of a stock is ± 0.03 (or a given threshold for that day) greater than or less than the open price to see if this will increase accuracy. The results were not promising, with all the models achieving accuracy below 50%. The results are shown in table 4. It was odd that the accuracy, precision, and recall scores were the same across each classifier.

4 Discussion

We realize that our results are not what we had hoped they would be. The best accuracy score happened with Google, with an accuracy score of 56.6%. While our results are less than promising, we feel that there is still a lot of improvement to be made.

Naive Bayes	Accuracy	51.6%
	Precision	45.9%
	Recall	52.9%
	F-Score	49.1%
Random Forest	Accuracy	51.0%
	Precision	48.8%
	Recall	52.0%
	F-Score	50.4%
SVM	Accuracy	49.5%
	Precision	30.0%
	Recall	50.7%
	F-Score	37.7%

Table 1: Results for MRK trained on 2333 headlines and tested on 667 headlines

Naive Bayes	Accuracy	50.3%
	Precision	65.2%
	Recall	57.0%
	F-Score	60.8%
Random Forest	Accuracy	53.2%
	Precision	63.1%
	Recall	59.9%
	F-Score	61.5%
SVM	Accuracy	56.6%
	Precision	87.7%
	Recall	59.0%
	F-Score	70.5%

Table 3: Results for GOOGL trained on 1105 headlines and tested on 316 headlines

Naive Bayes	Accuracy	52.9%
	Precision	79.5%
	Recall	53.1%
	F-Score	63.6%
Random Forest	Accuracy	52.0%
	Precision	50.6%
	Recall	54.0%
	F-Score	52.2%
SVM	Accuracy	48.6%
	Precision	0.6%
	Recall	54.1%
	F-Score	10.7%

Table 2: Results for MS trained on 2267 headlines and tested on 648 headlines

Naive Bayes	Accuracy	47.5%
	Precision	47.5%
	Recall	47.5%
	F-Score	47.5%
Random Forest	Accuracy	43.6%
	Precision	43.6%
	Recall	43.6%
	F-Score	43.6%
SVM	Accuracy	43.9%
	Precision	43.9%
	Recall	43.9%
	F-Score	43.9%

Table 4: Results for MRK using a threshold to choose between buy/hold/sell

There are many things that can be improved in order to model stock movements by measuring sentiment in news headlines. For example, one idea is to get an overall sentiment for a given day. There could be multiple news headlines all pertaining to the same stock on a given day. Further, these news headlines may not agree on the sentiment. One headline may be positive while another may be negative. It would be better to get an overall score for that day that would be used to train the model, which could result in more accurate predictions.

Furthermore, stock prices don't move much in a day. It may be more beneficial to look at overall sentiment for a week or month and see how the stock price changed in that time period. However, the disadvantage to an approach like this is that investors would like to predict stock movements in advance, and waiting a week to gather sentiment of headlines would render the model somewhat useless.

In addition, news headlines are not the only factor to stock movements. There are a lot of factors that contribute to overall stock movement that are not captured by our model. With that being said, it may be beneficial to look at other investments, such as cryptocurrencies, which may be more directly related to the news.

It would be interesting to look at classifiers such as an LSTM or RNN, where the outcome of the previous day would have an effect on the current day. Stock prices are somewhat dependent on the previous day's outcome so a model that can take that into account may be beneficial. However, our data set does not have headlines from every day in the date range which makes it difficult to train an LSTM or RNN.

Finally, it may not be possible for a linear model to predict this data. It would be interesting to look at diagrams of this data and see how the linear models fit the actual data. In doing so, it may be obvious that a non-linear model, such as a neural network, would do much better in predicting stock movements.

5 Project Reflection

Despite our project not achieving what we set out to achieve (high accuracy for predicting stock movements), I still believe that the project was a great learning experience. I enjoyed taking an idea and designing and developing a program that brings our idea to fruition with no outside help. Personally, I designed the classes for modeling stocks and fetching stock prices. I also built upon Jeremy's work with the FinBERT transformer to create a class that takes in all of our information and trains a classifier to use for prediction. Furthermore, I worked out the team contract. We all contributed to the codebase. The group contract was helpful but did not need to be used outside of the initial signing as team members adhered to the contract. I think it made meetings and communication easier as expectations were set from the beginning.

One of the most useful skills that I applied was debugging. There are a lot of things that can go wrong with a project, especially with no outside guidance. Being able to effectively debug the program led to quicker development. Another useful skill is documentation, which proved incredibly useful in a group project setting. This project certainly helped enhance my documentation skills as I was no longer writing documentation for myself but for other people to understand my code. While I would not say that it changed my way of documenting my code, it made me be more thorough with my documentation.

I feel that this project helped to enhance my collaboration skills as it required me to plan and design in a group. It also forced me to write well-documented, clean, and modular code due to the fact that other partners would be working on the same code.

If I could change anything, I would go back to the start of our project and choose a research paper that made all parts (or most) of their research publicly available such as code, data set, etc. This would have allowed us to better replicate results and build off of those results.

Limitations

One limitation is our corpora of news headlines, which are sparse and belong to only one news organization, which can introduce bias in actual sentiment of stock for the day. More data with more diverse news organizations may allow for a more accurate sentiment.

Acknowledgements

This document has been adapted by Nicholas DeBaise. The project has been completed in conjunction with Jeremy Perez, and Hawkeye Nadel.

(Araci, 2019) (Kalyani et al., 2016)

References

Dogu Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models.

Joshi Kalyani, Prof H. N. Bharathi, and Prof Rao Jyothi. 2016. Stock trend prediction using news sentiment analysis.