Predicting Stock Prices

Gan Jun Ying | Hon Jia Jing | Ryan Ang Jia Yong | Koh Rui En Colin | Zhang Xuan

Abstract

Traditionally, stock price movements are hard to predict. Our project aims to predict the direction of stock prices based on financial data and news headlines. As part of our research, we compare and evaluate different classification models while concurrently exploring different natural language processing techniques to utilize news headlines for model improvement.

1 Introduction

1.1 Motivation

In this modern age of increasing wealth, there has been a rise in stock trading as a means of earning money. Day traders make short term stock purchases in hopes to sell them quickly once the price increases. However, the volatility of the stock market makes it difficult to predict if a stock price will increase in the short term. We hope to empower these day traders to make data-driven decisions by using Machine Learning models that can predict short term stock price changes based on historical price movements and news headlines.

1.2 Guidelines

Given the stock data and relevant news headlines for an S&P 500 company, predict the direction of movement of the company's stock price 2 days after. Our research project revolves around experimenting with the effects of financial data and news headlines on classification accuracy of our models. Listed below are the research questions that we will address in this study:

1. *Which machine learning models are able to produce the best and most accurate predictions based on existing times-series data and news headlines?*
2. *Which features are more important in predicting changes to stock prices and how do we go about selecting them?*
3. *How should we approach the analysis of the headlines?*

2 Related Work

The ability to predict stock price movements on the financial market is highly sought after as it would offer a lucrative competitive edge for an investment entity over other stock market participants. Therefore, it is of no surprise that many academic researchers and industrial practitioners are drawn to this profitable topic. Many investment companies that have spent a huge sum of capital on IT infrastructures, data analytics and high frequency trading (HFT) systems in an attempt to predict the unforeseeable future of stock prices.

There has been a variety of technical approaches to making stock market predictions. Some such as *Pai and Lin (2005)* have used AutoRegressive Integrated Moving Average (ARIMA) models and Support Vector Machines [5], while others such as *Qian and Rasheed (2007)* used the ensembling of neural networks, decision trees, and KNN models [8]. Having seen the effectiveness of ensembling on stock movement predictions, our research project has utilized ensembling techniques to improve our models.

As technology advanced over the years, gathering news online has become effortless. With the overwhelming flow of readily accessible and available information, a topic of interest worth exploring would be the analysis of "headline risk" - the idea that news headlines or stories can influence the price of a stock, sector, or the broader market [1].

Many studies have been carried out to investigate the relationship between news headlines and stock price movements. *Ma (2008)* selected 29 stock names from Dow Jones Industrial Average and found news articles related to these stock names from the Wall Street Journal and Reuter Corpus. Instead of using the sentiments of these news articles to make predictions, he explored the usage of Naive Bayes and Maximum Entropy classifiers to directly predict the direction of the stock movement [4]. Similarly, our research mimics this idea by using word embeddings of news articles to directly predict the movement of stock prices.

Lee and Timmons (2007) believe that financial news articles play a role in influencing the movement of a stock. They point out that there exists some time lag between when the news article is released and when the market has moved to reflect this information [3]. We have approached our research project with this is mind by setting a 2 day prediction lookahead.

Many studies approach stock movement predictions from an "either or" perspective, where they either use only financial data or text related analysis for their predictions. By building on the research done by the aforementioned papers, we have constructed a model that utilizes both text analysis and financial data for predictions of stock movement.

3 Method, Evaluation & Discussion

3.1 Dataset

Our financial data is obtained from the Kaggle dataset (S&P 500 Stock Data), which contains the daily trading prices of the S&P 500 companies across a 5-year period (2013 to 2017). Our target class is a boolean value which represents whether the opening price of a stock 2 days later is higher than the current day's closing price. Upon further examination, we found that there is no class imbalance in the dataset. This is because our prediction lookahead is small and the stock market tends to be volatile in such a small time frame.

3.2 Data Collection

As this research project also explores the effect of news headlines on stock prices, we had to source for news headlines on the S&P 500 companies that coincided with the time frame of the dataset. We used the NewYorkTimes API to accomplish this. As an initial filter, we set the API search to return news headlines that tagged the relevant companies (eg. Facebook) as well as the child companies (eg. Instagram) - this filter returned news headlines or varying relevance. To further narrow down the relevance of the news, we checked to see if the returned news headlines and news snippets contained the respective companies' names.

3.3 Data Approach

The dataset we obtained contains data of all the individual S&P 500 companies. There were 2 ways we could approach the dataset:

1) *Train models using individual company data*
2) *Train models using data across multiple companies*

To decide which data approach best suited our use case, we compared the general classification accuracy of our classification models when we used individual company data to when we used data across multiple companies. The results were relatively poorer when data across multiple companies was used *(see Appendix B Figure 1a)*. We believe that the reason for the model's poor performance is because the stock prices of companies vary in different industries - each industry has their own unique characteristics. By using data across multiple companies, these characteristics were not properly captured and led to a poorer model. With that said, due to the lack of news data on individual companies, we had to use data across multiple companies when conducting our NLP research. In this scenario, we selected data from companies that were in the same industry (eg. Google, Facebook, Microsoft). *Appendix B Figure 1a* shows the slight performance improvement when using companies from the same industry as compared to companies from different industries.

3.4 Data Exploration

As part of our Data Exploration process, we used Time Series techniques to discover more insights about the trends and seasonal components of the respective company stock prices. By plotting the seasonal components of the data, we noticed that it had a weekly seasonal nature *(see Appendix A Figure 1)*. To further understand how stock prices depended on prices from previous days, we ran AutoArima on the data. AutoArima returns the optimal ARIMA model based on the minimization of the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) - in our case, it suggested an autoregressive (AR) component of order 1 for our ARIMA models. This meant that its stock prices only relied on stock data from the previous and nothing beyond. With a low dependence on historical data, we decided to focus our efforts into building accurate classification models rather than Time Series models. Considering that only the most recent days affect predictions, including data from previous days as features into the classification model seemed feasible.

3.5 Feature Engineering

In this section and the following section (Section 3.6 - Feature Selection), we address our research question *"Which features are more important in predicting changes to stock prices and how do we go about selecting them"*.

As part of improving our models, we came up with features to test with our models. Some of the features we engineered are as listed below:

1) *Moving Averages - 5 day, 20 day, 50 day*
2) *Varying the number of days of previous prices considered for predictions*
3) *Percentage returns: Percentage return on day n = (Stock price on day n - stock price on day n-1)/stock price on day n-1*

Moving Averages

Although Time Series models may not have been ideal for our use case, we wanted to integrate the benefits of Time Series data into our classification models. This meant that we had to curate features that contained the historical component of the stock prices. The first feature we engineered was the moving average. This feature is simply the average stock price over a certain historical time window from the current day. We experimented with different time windows and found that the combination of a 5-day, 20-day and 50-day window yielded the best results. By having information on moving averages, the model is equipped with the knowledge as to how the closing price of the day relates to these averages. Since stock markets tend to follow a certain trend, stock prices tend to stay within a certain standard deviation from the moving average.

Additionally, by having a combination of windows, the model may also be able to pick out additional information about how the stock prices are changing. For example, if there is a huge disparity between the 5-day moving average and the 50-day moving average, we can tell that there is a sudden spike in stock prices the past 5 days. With this information, it is likely that the price will be pulled in the direction of the 50-day moving average.

Varying the Number of Days for Previous-Day Data

When designing our input data, we varied the number of previous day data to include as input features, ranging from 1 (being just the data from the previous day) to 10 (data from 10 previous days). Choosing the number of days too low may provide insufficient data to train the classifier, whereas a number too high will blow up the input dimensionality which will result in overfitting. For each of our classification models, we experimented on the number of previous day data to be input as parameters *(see Appendix B Figure 1a)*. The optimal number of days is then selected based on the one which gave the highest average accuracy across the different classification models.

Percentage Returns

We used percentage returns instead of stock prices to more accurately account for the differences in returns across different companies. For instance, companies like Facebook have stock prices which lies in the range of 30 to 180, whereas companies like Google have stock prices in the range of 300 to 1200. In situations where we had to combine multiple datasets from different companies in our models, we introduced returns as a form of standardisation to measure the change in stock prices across the different companies.

Without using percentage returns, models such as K nearest neighbors may also not be able to correctly classify the movement of the stock price if the stock price falls outside the range of the training data seen by the model.

3.6 Feature Selection

As for our feature selection process, we wanted to ensure that the features we engineered were assisting the model in making better predictions. We used PCA to reduce the input dimensions to 2 PCA components and found that the features responsible for most of the variance were the volume and the 5-day moving average.

We also ran the training data through a Random Forest model to inspect the feature importance values generated by the model. As these feature importance values change due to the random nature of the algorithm, we ran the model multiple times and further cross checked with other feature selection algorithms. We saw that the moving averages were often among the top features:

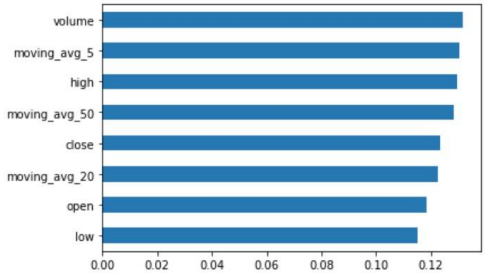

Figure 1. Feature importance values from Random Forest model.

These results gave us more confidence that the moving average features were assisting the model in making predictions. *Appendix B Figure 1* shows the actual improvements in model accuracy when using the moving average features. From the figure, we can see that moving averages were vital in building accurate machine learning models. The reason for this is likely because of the presence of traders that use technical analysis for purchasing stocks. These traders often rely heavily on financial charts and trends to make their purchase decisions - which explains the effectiveness of moving averages as an input feature.

3.7 Building Classification Models

In this section, we address our research question *"Which machine learning models are able to produce the best and most accurate predictions based on existing times-series data and news headlines"*.

We used accuracy as our metric to determine how well our model performs as there is no bias needed against false positives or negatives. At any point, the money earned from a successful classification will be equal to the money lost from a failed classification through buying or short selling stock. As such, accuracy will be a better indicator of the model's success when trading than a metric such as F-1 score, which penalises false classifications more severely. *Appendix B Figure 1* shows the experiments we conducted using various features and models. Model classification accuracies are validated using K-fold Cross Validation.

3.7.1 KNN Classifier

Performance of KNN with regards to the number of previous-day data:

In general, our KNN classifier performs better with increasing number of days (n) up till n = 3 *(see Appendix B Figures 1a & 1b)*. This can be intuitively explained by the fact that if a stock price has been increasing over a period of time, the returns of the stock over the past days would mostly be positive. Using a greater number of previous day data would allow the KNN classifier to account for the general direction of movement of stock prices across the past days, thereby possibly allowing for better classification accuracy. However, as mentioned, changes in direction of stock prices only rely on recent data - any data beyond the 3rd previous day loses its relevance.

Performance of KNN with regards to input dimensions:

With higher values of n, the KNN classifier performed better when we used PCA. We experimented with various number of reduced dimensions as seen in *Appendix B Figures 1a & 1b*. The KNN classifier is more sensitive to the curse of dimensionality as it uses the euclidean distance metric as a core part of its algorithm. With higher dimensions, more axes are created and distance between data points hold less significance. Thus, by performing PCA, the KNN classifier performs better.

3.7.2 Neural Network

In a Neural Network, the model is built by iteratively adjusting weights and biases so that the network can mimic the target function and make predictions from the input data. A Neural Network is a very powerful model that can fit very complex target functions, which leaves it susceptible to overfitting to the point that it effectively memorises the training data. This usually ends in poor performance when tested against a validation set.

The target function for stock data is extremely complex, and stock data is inherently noisy due to the fact that different investors buy and sell stocks for different reasons. This makes it challenging for fully connected Neural Networks to pick out the target function. As such, Neural Networks tend to overfit the stock data. Even after performing regularization techniques such as dropout regularization, the slight improvement in classification accuracy of the fully connected Neural Networks were not much different from simpler models such as Random Forest.

Even the LSTM Neural Networks did not perform as well as we hoped. The performance of the LSTM model is similar to the fully connected Neural Networks with dropout regularization. All results from our Neural Network experimentation can be found in *Appendix B Figures 2a & 2b*.

3.7.3 Ensembling the Ensembles

As seen from the *Appendix B Figure 1c*, the model with the highest classification accuracy is the ensemble model which ensembles the following models - Random Forest, AdaBoost and XGBoost. It achieved a consistent accuracy of around 60% when the input features were financial data and moving averages.

As part of our hyperparameter tuning process, we used GridSearch to select the optimal number of components for each of the models such that they individually performed optimally. AdaBoost and XGBoost did not show much improvement beyond 50 components, while Random Forest showed improvement up till about 200 components *(see Appendix B)*. All 3 predictors were generally in the range of 54-58% classification accuracy, with some fluctuation each time we ran Cross Validation.

In order to stabilize the classification accuracy, we aggregated the predictions of the 3 models and took a majority vote among them. The similar prediction accuracies between the models did not call for a weighted voting system.

It is important to note that the individual models have their own unique characteristics. Listed below are some of the key differences between the models:

Random Forest

- *Creates Decision Trees of varying heights that split on different features*
- *All Decision Trees have the same "voting power"*
- *Decision Trees are independently constructed*
- *Decision Trees have actual values as the leaf nodes*

AdaBoost

- *Creates Decision Stumps with different "voting power"*
- *Decision Stumps are built based on the error of the previous Decision Stump*
- *Decision Stumps have actual values as the leaf nodes*

XGBoost

- *Creates Decision Trees with a fixed number of leafs*
- *All Decision Trees have the same "voting power"*
- *Decision Trees are built based on the error of the previous Decision Tree*
- *Decision Trees have residuals as the leaf nodes*

As seen above, there are slight nuances between how the different models create their "forest" of Decision Trees. This uniqueness between the models is the reason why ensembling these ensemble models produce an effective classifier.

3.8 Building NLP Models

In this section, we address our research question *"How should we approach the analysis of the headlines".*

Data Preprocessing

In order to weigh the effects of news headlines on predictions, we decided to use only rows of data that contained news headlines. As the improvement in accuracy could potentially be very small, using only rows of data that contain news headlines will help us capture the difference in accuracy of a model that uses news headlines as compared to a model that does not.

Considering that news about a company does not occur every day, this meant that the volume of our training data decreased significantly. To account for this, we decided to use data from a subset of companies in the S&P 500 instead of using data from individual companies. As mentioned in Section 3.3, there is potential compromise in model performance when using data across multiple companies. In order to minimize this, we had to minimize the number of companies in the selected subset and the variation in industries between these companies. We strategically chose the 3 "Tech Giants" as our subset - namely Google, Microsoft and Facebook. Firstly, all 3 companies belong in the same industry - this minimizes the compromise on model performance that may be caused by the differing characteristics of stock prices in different industries *(see Appendix B Figure 1a)*. Additionally, these companies have significant impact on the Tech industry, which results to more news reports published on these companies - this helps with maximising volume of training data *(see Figure 2 below).*

Open	...	News	Company	Target
~~355.89~~			Google	~~True~~
356.63		Google reports increase in revenue	Google	True
...	...	...	...	...
~~124.54~~				~~False~~
123.91		Facebook acquires Instagram		True
...	...	...	...	...
110.88		Microsoft releases new software	Microsoft	True

Figure 2. Preprocessed dataset. Note that the rows without news are filtered out.

3.8.1 Word Embeddings with 1-Dimensional Convolutional Neural Networks

Building the Model

Once we had curated our data, we could move on to building our models. We used the pre-trained GloVe word vectors [6] to form our embedding layer. GloVe offers multiple options for the number of dimensions that represent the word vector - namely 50, 100, 200, 300. We found that the 100 dimension word vectors worked best for our use case *(see Appendix C Figures 1a & 1b)*.

As for the Neural Network, we used a 1-Dimension Convolutional layer with a kernel size of 4 which captures the 4-grams in the sentences, followed by a Pooling layer of size 3. These layers are repeated multiple times before finally being put through a fully connected layer. The output layer is a softmax layer.

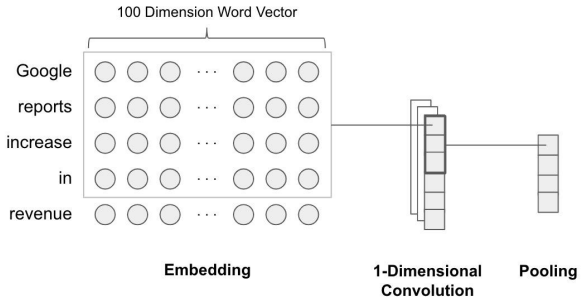

Figure 3. Visual representation of the Neural Network

We experimented with using different sizes for the convolutional layer and the pooling layer, but the aforementioned structure worked the best. In general, the model does not do particularly well in predicting the stock price direction *(see Appendix C Figures 1b & 1c for comparisons between different neural network structures).*

As the above experiment only used news headlines to make predictions, we wanted to see if financial data could be used alongside these word embeddings to improve our existing models. However, it was not trivial to input financial data into the Neural Network as the Embedding Layer only takes in words as inputs. As a workaround, we took the predictions from this Neural Network and fed it together with the financial data as an input feature into our existing classification models. Based on our results, the model still seems to perform poorly *(see Appendix B Figure 1c).*

3.8.2 Sentiment Analysis

Besides using Word Embeddings for predictions, we also experimented with using Sentiment Analysis. We used the VaderSentiment Library to calculate the compound sentiment scores for the news headlines. These compound sentiment scores have values from -1 to 1. We used financial data along with sentiment analysis for our classification.

Preprocessing

We experimented with using different preprocessing methods before feeding it as an input into the Vader Sentiment Analyzer. One such preprocessing step we did was to lemmatize the news headlines. Using lemmatised news headlines worked better than using raw headlines for sentiment analysis. Lemmatising the headline keeps the root form of all the words, which works better for stock analysis as it gives the same score for words which have the same root form. Words with the same meaning, no matter the word form, is likely to evoke similar emotions in the reader. This is why assigning the same score to similar words improves model accuracy *(see Appendix D Figure 1).*

Custom Sentiment Threshold

Investors in the stock market that do value investing tend to buy stocks based on the intrinsic value of the company such as net changes in cash

or dividends paid - these investors are investing for the long term and do not make hasty decisions to buy or sell based on trivial news headlines. To address this, we experimented with classifying compound sentiment scores into negative, neutral and positive sentiments based on our own custom threshold. A higher threshold will result in significant news headlines being classified as neutral, while a lower threshold will allow less significant news headlines to impact predictions. Through our experiments, we found that a threshold of 0.2 for positive sentiments and -0.2 for negative sentiments worked the best for our use case. The results are shown in the *Appendix D Figure 2*.

Using Historical Financial Data

Adding closing prices of previous days lowers the accuracy of predictions for days that have news headlines. Though historical data was beneficial for our classification models (eg. KNN) when we only used financial data, it does not appear to be the case when we include news headlines as a feature. It is likely that news headlines are a slightly better indicator of the direction of stock prices as compared to historical data. By adding historical financial data, it unnecessarily confuses the model. The results are shown in *Appendix D Figure 3*.

Using Historical News Sentiment

Adding the news headline sentiments of previous days also lowers the classification accuracy of our models. It seems that sentiments prior to the current day are nullified by the current day's sentiments. The results are shown in *Appendix D Figure 4*.

3.8.3 Enhancing Classification Models

Having seen optimistic results with our Sentiment Analysis models, we then integrated these features into our top performing model - the Ensemble of Ensembles. However, this new inclusion of features did not seem to improve the model. It is likely that the ratio of the number of data rows containing news headlines with respect to the total number of data rows in the entire dataset was not large enough to make a significant difference *(see Appendix A Figure 1c)*.

3.8.4 Initial Hypothesis VS Results

We initially hypothesized that the improvements from using news headlines would be more significant than the 3-5% improvement we observed in our experiments.

One possible reason for this may be because our prediction window is too short - we are predicting the direction of the opening price 2 days ahead. It seems likely that stock prices may take more than 2 days to stabilize after receiving a piece of news. If we had a longer prediction window (eg. 5 days ahead), the news headlines may have been more effective. With that said, we did not want to predict the direction of stock prices more than 2 days ahead as it causes a class imbalance. The stocks that we chose generally had an upward trend, meaning that the stock prices more than 2 days ahead would generally be increasing.

Another possible reason is that these news headlines affect investor decisions in more complex ways than we originally hypothesized. It is likely that the intricacy of investment decisions could not be fully captured with sentiment analysis.

4. Concluding Thoughts

Ensembling models are a very effective way of building a robust classifier that performs consistently well. As seen from our experiments *(see Appendix B Figure 1c)*, we managed to get slightly above 60% accuracy by ensembling Random Forest, AdaBoost and XGBoost - given the original financial data and feature engineered moving averages (5-day, 20-day, 50-day). On the other hand, Neural Networks seemed to underperform due to the noisy nature of the stock data.

With regards to analysis of news headlines, sentiment analysis with custom thresholds worked better than word embeddings. These news sentiments contribute to slightly better performance when used together with the original financial data. However, when used with our feature engineered moving averages, there were no conclusive improvements.

We believe that a key metric to a good Machine Learning model is its usability. Companies with lesser public influence tend to have fewer news reports published about them. In a practical sense, financial data is still the best approach to predicting the direction of stock prices. The classification models trained using financial data will still work effectively even when there are no news headlines on a company.

5. Further Exploration

Moving beyond the original scope of the project, we also explored how accurately Machine Learning models can predict actual stock prices.

5.1 Time Series Models

For our Time Series models, we trained our models using individual company data. We ran the data through the following Time Series models - ARIMA, SARIMAX and an LSTM Neural Network. These are common models used for Time Series regression problems.

5.1.1 ARIMA

At first glance of *Appendix E Figure 1*, the ARIMA model appears to be rather accurate. Upon further inspection, we notice that the predicted values are just shifted to the right. This does not make for a powerful model as it is naively predicting a based on the previous value.

5.1.2 SARIMAX

The SARIMAX is a Time Series model that can take in exogenous variables as input. Exogenous variables refer to input features apart from the historical target value. As seen in *Appendix E Figure 2*, the trained SARIMAX model seems to follow a similar general pattern as the actual values, but the predicted values seem to be rather far off.

5.1.3 LSTM Neural Network

We built an LSTM Neural Network using Keras and trained the model using different inputs. The results of using only the closing price is shown in *Appendix E Figure 3*. The results of using all original input features is shown in *Appendix E Figure 4*.

6. Future Work

While the prediction accuracy of our model is satisfactory, our project could be further developed by conducting more complex feature engineering based on financial domain expertise, as well as using more advanced machine learning techniques.

As for sentiment analysis, examining the association of news headlines between two companies may be something worth looking into. We could extract sentiments on a particular company and analyse if it impacts other companies over time.

Additionally, extracting news headlines from different news sources and combining these headlines together to carry out prediction may lead to a higher accuracy. Since news articles are often subjective and opinionated for each news source, selecting from a variety of news sources would increase the sample size and resulting in less bias. This may be a more accurate representation of the news landscape surrounding a stock.

We are also keen to explore how the Unsupervised Sentiment Neurons system [9] may affect our model predictions. This system classifies text based on the plutchik wheel of emotions [7]. This may have given us more insight to the news headlines and improved our model accuracy.

7. References

[1] Chen, J. (2019). Headline Risk. Retrieved from https://www.investopedia.com/terms/h/headline-risk.asp

[2] Kirange, D. K., & Deshmukh, R. R. (2016). Sentiment Analysis of News Headlines for Stock Price Prediction. Retrieved from https://www.researchgate.net/profile/D_Kirange/publication/299536363_Sentiment_Analysis_of_News_Headlines_for_Stock_Price_Prediction/linksz/56fe2ffa08aea6b774683eb5/Sentiment-Analysis-of-News-Headlines-for-Stock-Price-Prediction.pdf?origin=publication_detail

[3] Lee, K., & Timmons, R. (2007). Predicting the Stock Market with News Articles. Retrieved from https://nlp.stanford.edu/courses/cs224n/2007/fp/timmonsr-kylee84.pdf

[4] Ma, Q. (2008). Stock Price Prediction Using News Articles. Retrieved from https://nlp.stanford.edu/courses/cs224n/2008/reports/21.pdf

[5] Pai, P.-F., & Lin, C.-S. (2005). A hybrid ARIMA and support vector machines model in stock price forecasting

[6] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. Retrieved from https://nlp.stanford.edu/projects/glove/

[7] Plutchik's Wheel of Emotions. (n.d.). Retrieved from https://www.6seconds.org/2017/04/27/plutchiks-model-of-emotions/

[8] Qian, B., & Rasheed, K. (2006). Stock market prediction with multiple classifiers. doi: 10.1007/s10489-006-0001-7

[9] Radford, A., Sutskever, I., Józefowicz, R., Clark, J., & Brockman, G. (2017). Unsupervised Sentiment Neuron. Retrieved from https://openai.com/blog/unsupervised-sentiment-neuron/

8. Appendix

Appendix A: Time Series Data Exploration

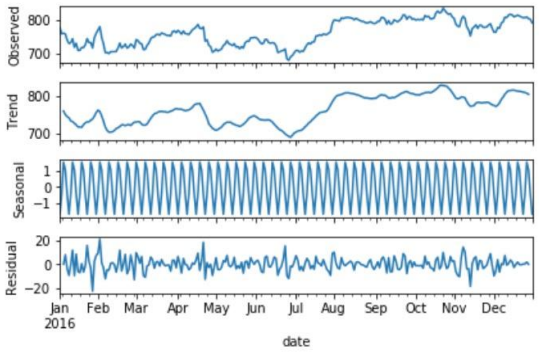

Figure 1. Time Series analysis of Google's financial data

Appendix B: Classification Model Performance

Input	Financial Data	Financial Data	Financial Data	Financial Data + Financial Data 1 day before	Financial Data + Financial Data of 3 previous days
Company	Google	Google, Berkshire Hathaway, Johnson & Johnson	Google, Facebook, Microsoft	Google	Google
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
KNN Classifier (K = 3)	54.1%	52.0%	52.2%	53.0%	54.3%
KNN Classifier (K = 7)	54.0%	52.9%	53.2%	54.0%	54.2%
KNN Classifier (K = 15)	54.2%	52.6%	53.2%	53.4%	54.0%
Random Forest (n = 50)	53.1%	52.5%	52.5%	53.2%	53.5%
Random Forest (n = 100)	53.5%	52.6%	53.2%	53.4%	53.1%
Random Forest (n = 200)	54.1%	52.7%	53.5%	54.4%	54.1%
Random Forest (n = 300)	53.8%	53.1%	53.6%	53.9%	54.2%
AdaBoost (n = 10)	55.0%	53.8%	53.9%	54.7%	54.2%
AdaBoost (n = 50)	55.9%	54.5%	54.4%	54.9%	55.6%
AdaBoost (n = 100)	55.5%	54.6%	54.2%	55.6%	55.8%
XGBoost (n = 10)	56.1%	53.2%	54.5%	55.8%	55.6%
XGBoost (n = 50)	57.4%	54.1%	55.1%	57.2%	56.6%
XGBoost (n = 100)	56.5%	53.7%	55.1%	55.9%	56.5%
Ensemble (Random Forest, AdaBoost, XGBoost)	57.2%	54.3%	55.5%	57.0%	56.9%

Figure 1a. Classification accuracy of various models ran over different features

Input	Financial Data + Financial Data up to the 3rd previous day + PCA to 5 dimensions	Financial Data + Financial Data up to the 3rd previous day + PCA to 10 dimensions	Financial Data + Financial Data up to the 5th previous day + PCA to 5 dimensions	Financial Data + Financial Data up to the 5th previous day + PCA to 10 dimensions
Company	Google	Google	Google	Google
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
KNN Classifier (K = 3)	54.0%	54.2%	53.1%	53.9%
KNN Classifier (K = 7)	54.7%	54.8%	54.3%	54.7%
KNN Classifier (K = 15)	54.3%	54.2%	54.0%	53.8%
Random Forest (n = 50)	53.3%	53.8%	53.4%	53.2%
Random Forest (n = 100)	53.4%	53.9%	53.2%	53.5%
Random Forest (n = 200)	55.1%	55.5%	54.0%	55.3%
Random Forest (n = 250)	54.8%	55.2%	53.9%	54.9%
AdaBoost (n = 10)	54.9%	55.3%	54.3%	55.1%
AdaBoost (n = 50)	56.4%	56.3%	55.2%	56.5%
AdaBoost (n = 100)	56.1%	56.4%	55.3%	56.3%
XGBoost (n = 10)	55.7%	55.9%	55.5%	55.4%
XGBoost (n = 50)	56.5%	56.4%	56.2%	56.4%
XGBoost (n = 100)	55.7%	55.0%	56.6%	55.9%
Ensemble (Random Forest, AdaBoost, XGBoost)	56.6%	57.0%	56.7%	56.8%

Figure 1b. Classification accuracy of various models ran over different features

Input	Financial Data + Moving Averages (5, 20, 50)	Financial Data + Moving Averages (5, 20, 50)	Financial Data + Financial Data 1 day before + Moving Averages (5, 20, 50)	Financial Data + Moving Averages (5, 20, 50) + Sentiment Scores with Custom Threshold	Financial Data + Moving Averages (5, 20, 50) + Word Embedding Predictions
Company	Google	Google, Facebook, Microsoft	Google	Google	Google
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
KNN Classifier (K = 3)	55.1%	55.2%	54.3%	54.9%	52.4%
KNN Classifier (K = 7)	56.1%	55.0%	55.4%	55.3%	53.3%
KNN Classifier (K = 15)	55.8%	55.2%	54.2%	54.8%	53.6%
Random Forest (n = 50)	54.3%	54.1%	54.5%	54.4%	52.3%

Random Forest (n = 100)	55.2%	54.6%	55.1%	55.0%	53.4%
Random Forest (n = 200)	56.5%	55.9%	56.2%	56.6%	53.5%
Random Forest (n = 250)	56.2%	55.9%	56.0%	56.0%	53.9%
AdaBoost (n = 10)	56.1%	56.2%	55.3%	56.2%	53.8%
AdaBoost (n = 50)	56.5%	55.6%	56.6%	56.4%	54.5%
AdaBoost (n = 100)	56.4%	55.7%	56.7%	56.8%	54.2%
XGBoost (n = 10)	56.9%	56.3%	56.5%	57.0%	54.8%
XGBoost (n = 50)	58.1%	56.9%	57.0%	58.4%	55.5%
XGBoost (n = 100)	57.9%	56.2%	57.1%	57.3%	54.9%
Ensemble (Random Forest, AdaBoost, XGBoost)	**60.1%**	56.9%	57.4%	59.9%	55.9%

Figure 1c. Classification accuracy of various models ran over different features

Input	Financial Data	Financial Data + Financial Data up to the 3rd previous day	Financial Data + Financial Data up to the 3rd previous day + PCA to 10 dimensions
Company	Google	Google	Google
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Neural Network (4 hidden layers with 10 neurons each)	53.4%	53.2%	53.1%
Neural Network (Dropout layer + 4 hidden layers with 10 neurons each)	55.1%	54.9%	54.5%
Neural Network (Dropout layer + 4 hidden layers with 50 neurons each)	54.9%	54.6%	55.2%
Neural Network (Dropout layer + 10 hidden layers with 10 neurons each)	55.3%	55.0%	55.2%
Neural Network (3 LSTM layers with 5 units each)	55.2%	54.7%	54.8%
Neural Network (7 LSTM layers with 5 units each)	55.5%	54.6%	55.0%
Neural Network (3 LSTM layers with 20 units each)	54.8%	54.9%	54.6%

Figure 2a. Classification accuracy of Neural Networks ran over different features

Input	Financial Data + Moving Averages (5, 20, 50)	Financial Data + Moving Averages (5, 20, 50)	Financial Data + Financial Data 1 day before + Moving Averages (5, 20, 50)	Financial Data + Moving Averages (5, 20, 50) + Sentiment Scores with Custom Threshold
Company	Google	Google, Facebook, Microsoft	Google	Google
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Neural Network (4 hidden layers with 10 neurons each)	55.1%	54.9%	55.3%	54.8%
Neural Network (Dropout layer + 4 hidden layers with 10 neurons each)	56.2%	55.3%	55.8%	55.7%
Neural Network (Dropout layer + 4 hidden layers with 50 neurons each)	55.8%	55.1%	55.5%	56.0%
Neural Network (Dropout layer + 10 hidden layers with 10 neurons each)	56.1%	55.7%	55.5%	**56.2%**
Neural Network (3 LSTM layers with 5 units each)	56.1%	-	55.1%	56.2%
Neural Network (7 LSTM layers with 5 units each)	56.2%	-	55.4%	55.6%
Neural Network (3 LSTM layers with 20 units each)	55.6%	-	54.9%	55.6%

Figure 2b. Classification accuracy of Neural Networks ran over different features

Appendix C: Word Embedding Model Performance

Input	News Headlines	News Headlines	News Headlines
Company	Google, Facebook, Microsoft	Google, Facebook, Microsoft	Google, Facebook, Microsoft
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Embedding Dimension	50	100	200
Model	3 layers of 1-D Convolution of size 3, interleaved with Pooling layers of size 3	3 layers of 1-D Convolution of size 3, interleaved with Pooling layers of size 3	3 layers of 1-D Convolution of size 3, interleaved with Pooling layers of size 3
Accuracy	53.2	53.7	53.2

Figure 1a. Classification accuracy of GloVe Word Embeddings ran over different Neural Network structures

Input	News Headlines	News Headlines	News Headlines
Company	Google, Facebook, Microsoft	Google, Facebook, Microsoft	Google, Facebook, Microsoft
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Embedding Dimension	300	100	100
Model	3 layers of 1-D Convolution of size 3, interleaved with Pooling layers of size 3	3 layers of 1-D Convolution of size 3, interleaved with Pooling layers of size 4	3 layers of 1-D Convolution of size 4, interleaved with Pooling layers of size 3
Accuracy	53.3	53.4	**54.1**

Figure 1b. Classification accuracy of GloVe Word Embeddings ran over different Neural Network structures

Input	News Headlines	News Headlines	News Headlines
Company	Google, Facebook, Microsoft	Google, Facebook, Microsoft	Google, Facebook, Microsoft
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Embedding Dimension	100	100	100
Model	3 layers of 1-D Convolution of size 4, interleaved with Pooling layers of size 4	3 layers of 1-D Convolution of size 5, interleaved with Pooling layers of size 3	3 layers of 1-D Convolution of size 5, interleaved with Pooling layers of size 5
Accuracy	53.7	52.9	53.5

Figure 1c. Classification accuracy of GloVe Word Embeddings ran over different Neural Network structures

Appendix D: Sentiment Analysis Model Performance

The opening price, closing price, highest price, lowest price and volume of trade are used as inputs for all models.

Input	Raw, classified sentiments	Raw, exact sentiments	Lemmatised, classified sentiments	Lemmatised, classified sentiments
Company	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
KNN classifier	57.57	52.73	**57.88**	54.34
Random Forest	53.58	52.73	58.20	54.66
Adaboost	53.69	52.73	54.98	51.45

Figure 1. For classified sentiments. Threshold: sentiment values < -0.1 are classified as negative sentiments, sentiment values > 0.1 are classified as positive sentiments.

Input	Classified, threshold = (<-0.1, >0.1)	Classified, threshold = (<-0.2, >0.2)	Classified, threshold = (<-0.25, >0.25)	Classified, threshold = (<-0.3, >0.3)
Company	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Random Forest	57.88	**59.48**	56.27	52.09

Figure 2. Lemmatised news data. Thresholds are specified in the Input row (eg. (<-0.1, >0.1) means that sentiment values < -0.1 are classified as negative sentiments, sentiment values > 0.1 are classified as positive sentiments

Input	Closing price of current day only	Closing price of current day & 1 day before	Closing price of current day & 3 days before	Closing price of current day & 5 days before
Company	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Random Forest	**59.48**	53.38	55.95	58.20

Figure 3. Lemmatised news data. Threshold: sentiment values < -0.2 are classified as negative sentiments, sentiment values > 0.2 are classified as positive sentiments.

Input	Headline of current day only	Headlines of current day & 1 day before	Headlines of current day & 3 days before	Headlines of current day & 5 days before
Company	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook	Google, Microsoft, Facebook
Target	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead	Direction of opening price 2 days ahead
Random Forest	**59.48**	53.05	53.70	52.73

Figure 4. Lemmatised news data. Threshold: sentiment values < -0.1 are classified as negative sentiments, sentiment values > 0.1 are classified as positive sentiments.

Appendix E: Time Series Regression

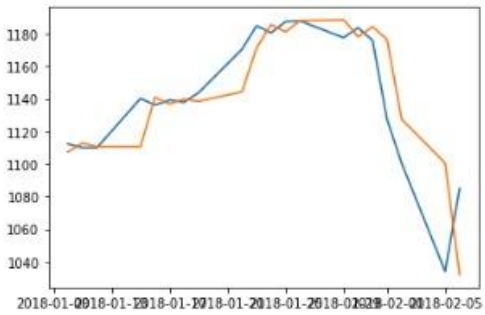

Figure 1. Predictions of the ARIMA model, Order(1, 1, 0). Blue plot represents the actual values. Orange plot represents predicted values.

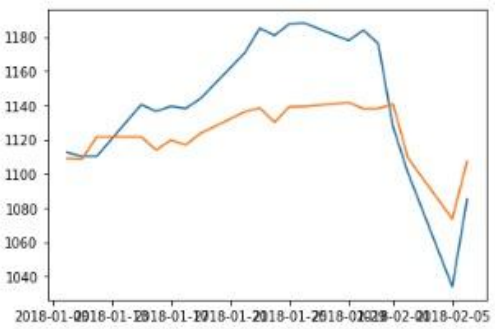

Figure 2. Predictions of the SARIMAX model, Order(0, 1, 0). Blue plot represents the actual values. Orange plot represents predicted values.

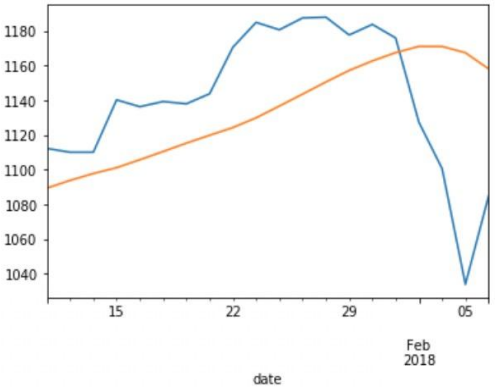

Figure 3. Predictions of the LSTM Neural Network (using only historical closing prices). Blue plot represents the actual values. Orange plot represents predicted values.

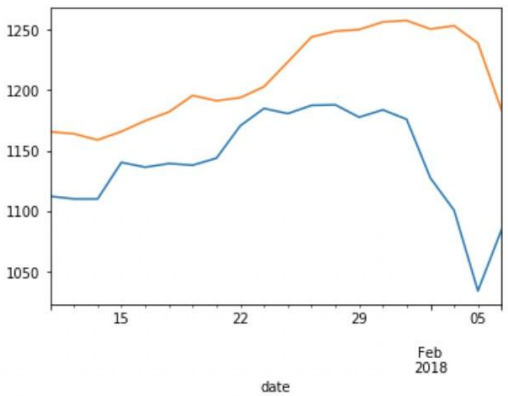

Figure 4. Predictions of the LSTM Neural Network (using all input features). Blue plot represents the actual values. Orange plot represents predicted values.