# Prediction of Stock Market Movement with Sentiment Analysis

CS3244-2010-0035

Jess Teo Xi Zhi (e0309463) | Benedict Chua Song Shing (e0199458) | Joshua Tam Wei Ian (e0325893)
Wong Chuan Kai (e0308994) |Lim Jing Xiang (e0415682) | Leow Chen Yue, Abel (e0425930)

## Abstract

In the current era, the movement of the stock market is greatly impacted by news reports. They influence the opinions and decisions of shareholders, which impacts the stock market prices. In this report, we examine the correlation between news sentiments and stock prices and analyse the accuracy of the different models trained to predict the shift in these prices based on the sentiments.

The sentiments of top news articles each day is determined using Natural Language Processing (NLP), and the impact of these sentiments are measured. Valence Aware Dictionary for Sentiment Reasoning (VADER) was used as the NLP model for determining the sentiment of each news article. The sentiment scores are merged with the SPDR S&P 500 ETF Trust fund (SPY) stock prices and are used to train our models.

Currently, the model which gives the best accuracy is the Naive Bayes model, with 57% accuracy. This matches the standard achieved by other models presented on Kaggle.

## Introduction

The analysis of stock trends is an interesting area of research as there are currently no universal rules or guidelines in predicting stock market movement. With the ease of accessibility of news reports, the content within these articles can easily influence an investor's opinions and actions within the stock market, which may affect the shift of the stock market.

Many previous works and research have suggested a relationship between news sentiments and stock returns. Generally, media sentiments have been found to be an important predictor of daily stock returns. An increase in optimistic news predicts a short run increase in stock returns while pessimistic news was associated with a temporary decrease in stock returns. This is further elaborated in the next section.

Information plays an integral part in our project as our project is primarily based on news articles. As such, we formulated a set of questions before starting our project that served as our goals.

1. Do positive news articles generally lead to an increase in stock prices?
2. And do negative news articles generally lead to a fall in these prices?
3. Are we able to predict whether our stocks will rise or fall by the sentiments of news articles?

We will address these questions again in the rest of the report.

With these goals in mind, the primary focus of our project is to predict the stock market movement using the sentiment scores of news articles generated by VADER model. We used the top news articles of each day on the assumption that these articles are popular and have more impact than the other news articles.

In the next few sections, we described other works and research relevant to our project before elaborating more on the datasets we used and our methodologies. We then provide our analysis on our findings and further discussions on our project before concluding our report.

## Related Work

As mentioned in the previous section, there were many related research and works done previously. Below is a list of research and algorithms that were implemented and proposed to predict the price of stock.

Hao Yee and Anthony [7] built a model that aims to predict the movements of NASDAQ in the immediate future given a news dataset. They provided a well-grounded framework that allows us to have a basic foundation to begin with. They trained and compared the accuracies using these models: Multinomial Naive Bayes (MNB), Gaussian Discriminant Analysis (GDA), and Support Vector Machines (SVM) and obtained a 65% accuracy in both SVM and NMB. Their models were trained using 1 year worth of news dataset. From their research project, we decided to have more news dataset as 1 year worth of news data might be insufficient to obtain a well-trained model. The lack of data may have affected the accuracy of their models.

In another related work, daily tweets' sentiments from twitter were analysed and used to predict the trend of the stocks, specifically on DJIA closing values [8]. This project is based on neural networks (self-organising fuzzy) and their studies concluded that ANN has a better forecast accuracy than time series model. However, after analysing their research paper, we noticed that their model only performs better at known patterns but fails to perform at unknown patterns.

Finally, we also studied the following project on stock price predictions based on full-text articles and its news summary [9]. Full text articles can gather both useful information and noise that are irrelevant to the subject matter. As such, this research focuses mainly on the pre-processing of data in a typical news-stock prediction work. Their summarised models generally perform better than full text one. However, if their stock information covers only a few news, then the summary might not be good as the summarised text might have resulted in less emphasis on the relevant stocks, hence, lowering the quality of the news dataset.

We identified the weaknesses in each of the related work above and attempted to address them in our project.

## Dataset

Our dataset consists of 2 separate sets of news articles and 1 set of stock history prices. We combined these 3 datasets together in order to achieve a more current and diverse dataset. Our first dataset was obtained from Kaggle for preliminary training and testing. The other news dataset was scraped from Reddit to retrieve more recent news articles for further testing. Finally, we retrieved our third set, SPY's stock history, from Yahoo Finance.

As the SPDR S&P 500 ETF Trust fund (SPY) is the largest fund in the world, our team believed it would be a good representation of other stocks that follow the movement of SPY. As such, our models will be trained to predict the movements of the SPY stock prices in the immediate future.

The reason for having 2 sets of news headline data was because the initial Kaggle dataset only contains the top 25 news headlines daily from reddit from 2008 to 2016 and the DJIA stock data for the same time period. As this dataset contains irrelevant data and is not up to date, we did some data cleaning to remove the DJIA stock data and scraped our second dataset from Reddit which contains the top 25 news articles from 2016 to 2020 to include the most latest news. Lastly, we obtained the stock data for S&P 500 SPY from Yahoo Finance, for the respective years from 2008 to 2020 to match our news headline dataset. We then combined all 3 datasets and cleaned them to obtain a cleaned data set from 2008 to 2020 (3072 days).

### Data Cleaning and Feature Extraction

As News Headlines may come in different formats, and include special symbols, we first cleaned our data by doing the following.

1. Converting the headlines to lowercase letters
2. Removing the HTML tags using regex
3. Manually filled the days with missing news headlines with relevant news

We chose not to remove punctuations as VADER is able to use these information to better determine VADER sentiment score. Next, we generated a single string combining the Top 25 news headlines and store them in a new column called "Combined_News" for each day, which are used as inputs for sentiment analysis subsequently.

## Our Methodology

To predict the closing price for the next day using the current day's set of Top 25 Headlines, we generated a Label column, labelling a day as "1" if the price of SPY stock increased or stayed the same on the next day, and "0" if the price of SPY stock decreased the next day. These labels were generated using the SPY stock dataset by deducting the adjusted close price of the next day with the price of the current day.

$$Label = \begin{cases} 1 & if \ \ stock\ increases\ or\ same \\ 0 & if \ \ stock\ decreases \end{cases}$$

Once the labels were generated, we used a 2-step approach to tackle the problem. This involves breaking News Headlines down into their sentiment scores, and subsequently using supervised learning machine learning techniques to train on those data.

### 1. Sentiment Analysis

Sentiment analysis (or opinion mining) uses natural language processing and machine learning to interpret and classify emotions in subjective data. Sentiment analysis can detect polarity (i.e. a positive or negative opinion) within a text and can be done on a whole document, a paragraph, a sentence, or a clause.

Understanding people's emotions is essential for the stock market since opinions and thoughts are being expressed more openly on the news and internet than ever before. This is why we have chosen to go with a "Emotion Detection" type of Sentiment Analysis, that aims at detecting emotions, like happiness, frustration, anger, sadness, and so on. As many emotion detection systems use lexicons, such as a list of words and the emotions they convey, we found that both NLTK's Valence Aware Dictionary and Sentiment Reasoner are great open-source packages that can help us achieve this task.

TextBlob is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

VADER or Valence Aware Dictionary and Sentiment Reasoner is a rule/lexicon-based, open-source sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains.

It contains a comprehensive dictionary of not only words, but also phrases (e.g 'too much'), emoticons (e.g. ':-)'), and sentiment-laden acronyms (e.g. 'OMG', 'GG'). All the lexical features were rated for polarity and intensity on a scale from '-4: Extremely Negative' to '+4: Extremely Positive'.

In our project, we used TextBlob to give us the subjectivity and polarity of the text, and Vader Sentiment to obtain the sentiment scores (i.e. negative, positive, neutral and compound), for each day.

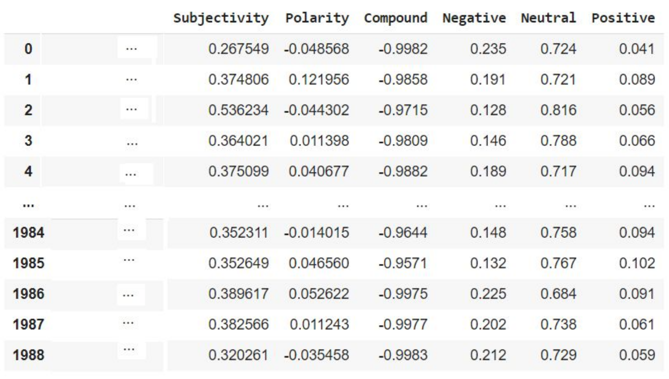
Below is a list of the sentiment scores and its definition that are generated by VADER when news headlines are passed into it.

- **Subjectivity** refers to how opinionated the text is. [0,1]
- **Polarity** refers to how opinionated the text is. [-1,1]
- **Positive** and **Negative** would show the positive and negative sentiments of the text respectively. [0,1]
- **Neutral** would show the neutrality score of the text. [0,1]
- **Compound** is the normalized score computed based on some heuristics and sentiment intensity. [-1,1]

$$Compound\ Score = \frac{\sum Sentiment\_value}{\sqrt{(\sum Sentiment\_value)^2 + 15}}$$

Capitalisation, degree modifiers, polarity shift due to conjunctions and catching polarity negation are examples of heuristics.

1. Capitalisation (especially all-caps) increases the magnitude of sentiment intensity without affecting semantic orientation.
2. Degree modifiers change the magnitude of sentiment intensity slightly based on the modifiers used.
3. Polarity shift in conjunctions will have a shift in the sentiment polarity based on conjunction used so the sentiment score often has mixed sentiments..
4. Catching polarity negation where there are multiple negations in a given text.



| | | Subjectivity | Polarity | Compound | Negative | Neutral | Positive |
|---|---|---|---|---|---|---|---|
| **0** | ... | 0.267549 | -0.048568 | -0.9982 | 0.235 | 0.724 | 0.041 |
| **1** | ... | 0.374806 | 0.121956 | -0.9858 | 0.191 | 0.721 | 0.089 |
| **2** | ... | 0.536234 | -0.044302 | -0.9715 | 0.128 | 0.816 | 0.056 |
| **3** | ... | 0.364021 | 0.011398 | -0.9809 | 0.146 | 0.788 | 0.066 |
| **4** | ... | 0.375099 | 0.040677 | -0.9882 | 0.189 | 0.717 | 0.094 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **1984** | ... | 0.352311 | -0.014015 | -0.9644 | 0.148 | 0.758 | 0.094 |
| **1985** | ... | 0.352649 | 0.046560 | -0.9571 | 0.132 | 0.767 | 0.102 |
| **1986** | ... | 0.389617 | 0.052622 | -0.9975 | 0.225 | 0.684 | 0.091 |
| **1987** | ... | 0.382566 | 0.011243 | -0.9977 | 0.202 | 0.738 | 0.061 |
| **1988** | ... | 0.320261 | -0.035458 | -0.9983 | 0.212 | 0.729 | 0.059 |

Fig 1: Sentiment scores of news dataset

### 2. Training our Model

After generating the sentiment scores of the headlines for each day, we merged the sentiments dataset with the SPY stocks dataset. The sentiments score datasets were merged with the next day stock prices in the SPY stocks dataset. This is to provide a timeframe for which the sentiments of the news come into effect and affect the stock prices.

The combined dataset will then be used to train our models. The models we trained on are listed below.

- ➢ Gaussian Naive Bayes
- ➢ K-Nearest Neighbours
- ➢ Decision Tree Classifier
- ➢ Logistic Regression
- ➢ Random Forest Classifier
- ➢ Extreme Gradient Boosting (XGBoost)
- ➢ Linear Discriminant Analysis
- ➢ Support Vector Classifier

Below is a description of Gaussian Naive Bayes, Extreme Gradient Boosting and Linear Discriminant Analysis. We will only be describing these classifiers as the other models were taught in class.

**Gaussian Naive Bayes**

The model with the best accuracy came from the Gaussian Naive Bayes model. It is a probabilistic classification method, and attempts to classify the features into groups based on Bayes Theorem. Gaussian Naive Bayes supports continuous valued features and models each feature as a Gaussian distribution. The features are assumed to be independent of each other, thus considers them separately when determining the final probability.

Let $w_1$ be the class label of 1 that indicates the stock price went up or remain the same and $w_2$ to be the class label of 0 which indicates that the stock price decreased.

$$P(w_i \mid x) = \frac{p(x \mid w_i)\,P(x_i)}{\sum_{i=1}^{2} p(x \mid w_i)\,P(w_i)}.$$

For a binary classification, we have two Bayes classification rules as shown below:

If $P(w_1|x) > P(w_2|x)$, $x$ will be classified as $w_1$

If $P(w_2|x) > P(w_1|x)$, $x$ will be classified as $w_2$

Using the Gaussian probability distribution function, we are able to use $\mu_i$ as the mean value and $\Sigma_i$ as the covariance matrix to solve for the discriminant functions.

$$p(x \mid w_i) = \frac{1}{(2\pi)^{n/2}\left|\Sigma_i\right|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right) \quad i = 1, 2.$$

The data can be separated by a hyperplane, if they have an equal covariance matrix, and a hyperquadrics, if the covariance matrix is unequal.

**XGBoost**

XGBoost (eXtreme Gradient Boosting) is a software library which provides a gradient boosting framework for multiple languages. It implements the gradient boosting tree decision algorithm, and contains other key algorithms to automatically handle missing data, support parallelisation of tree construction, and allow for continued boosting of an already fitted model with new data. It supports both regression and classification predictive modelling problems.

The implementation of Gradient Boosting Decision Tree is similar to AdaBoost with an exception that Gradient Boost trees have a depth larger than 1. It uses an ensemble tree methods by boosting weak learners using the gradient descent architecture.

Compared to the original Gradient Boosting algorithm above, XGBoost includes regularisation that penalises more complex models and 'learning' the best missing value based on training loss. The algorithm also comes with a built-in cross-validation at each iteration to find the best hyperparameters without the need to explicitly program the validation.

Optimal split points among weighted datasets are determined using the distributed weighted Quantile Sketch Algorithm. [10]

**Linear Discriminant Analysis (LDA)**

Linear Discriminant Analysis (LDA) is used as a tool for classification, dimension reduction and data visualisation. In our project, we will be using it as a classification model.

LDA assumes that the data is Gaussian and each feature has the same variance. Under these assumptions, the model estimates the mean and variance of the dataset for each feature.

It estimates the probability that a new set of inputs belong to each class and make predictions based on these probabilities. The input will be classified under the class with the highest probability. The probabilities of each class are calculated using a discriminant function.

$$\delta_k(\mathbf{x}) = \mathbf{x}^T \boldsymbol{\Sigma}^{-1}\mu_k - \frac{1}{2}\mu_k^T \boldsymbol{\Sigma}^{-1}\mu_k + \log \pi_k$$

As the name suggests and from the equation above, we observe that the discriminant function is a linear function under the assumption that the variance for each class is the same.

### 3. Evaluating our Results

In our project, one of the objectives is to reduce the number of false positives and false negatives in our results. As both of them are equally consequential in this project, they are assigned the same weight as we aimed to push their counts as low as possible.

We will also be measuring using their Accuracy / Precision / Recall / F1 score. Using these measures, we can compare the accuracy of all the models and determine the model with the highest accuracy.

## Evaluation

### 1. Dataset

As mentioned in the earlier section, our initial dataset was obtained from Kaggle, and contained a combination of daily top 25 news articles from 2008 to 2016 and similar labels for the DJIA Stock Market movement (0 for decrease, and 1 for increase or no change). This dataset was initially used for our preliminary experimentation and training of various models. However, we had 2 main issues with this initial dataset. Firstly, it was not up-to-date as there were only news reports till 2016. The second issue was that DJIA only tracks 30 stocks, which is a miniscule part of the market. This may also give some listed companies unfair advantages, resulting in an inaccurate representation.

Hence, our team researched and scraped for a more improved dataset. This consisted of the updated news headlines to the current year, 2020. Since the initial Kaggle dataset had sourced its news headlines from Reddit, we similarly scrapped for more recent daily top 25 news articles from Reddit to ensure there was no data contamination. These were combined with the articles obtained from Kaggle to provide us a more up-to-date dataset.

Our target stock market was SPDR S&P 500 ETF Trust fund (SPY) as it is the largest fund in the world, comprising 505 stocks, in which they include the 30 found within DJIA. This solved the issue revolving around using DJIA and provided us with a better representation of other similar stock markets.

### 2. Baselines (Show why the baselines are legitimate targets for comparison/ delete later)

For our project, we have chosen existing kaggle competition notebook results under a similar scope, to serve as our baselines for comparison. The methods and models used in those notebooks have an average accuracy of around 50-54% with the highest accuracy being around 57%. They were also trained using supervised learning machine learning techniques.

As the methods and models used in Kaggle competitions are contributed by the community, it would be a good target for comparison in our project where we aimed to achieve an accuracy higher than that in the competition..

Hence, the goal for our project was to gain a deeper understanding on how and why the models perform the way they did, the different factors and limitations within the models presented, and tried to replicate and improve from there with our own model and curated dataset.

Our modest goal was to achieve above 60% accuracy, or attain a model that was at least as accurate as the well-performing ones seen on kaggle. That would serve as success for our project.

## 3. Main Experimental Results

In our original preliminary result, we obtained accuracies of about 80-82% using LDA and XGBoost. This was measured using the Receiver Operating Characteristic (ROC) Curve. The results were plotted in Fig. 3 below.

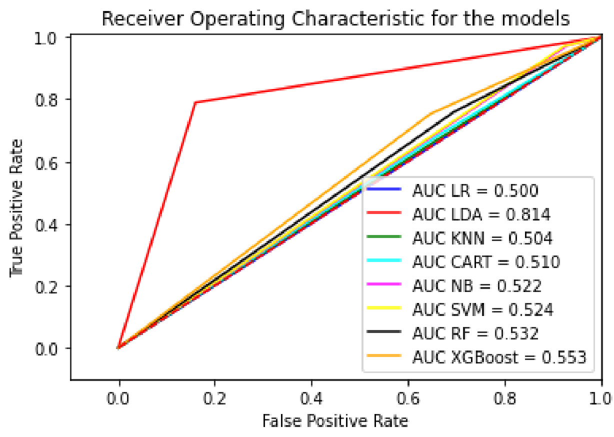


Fig 2: Preliminary results ROC curve for all models

In Fig. 2, we can observe that the accuracy for LDA is well above the other models. In addition, we proceeded to perform feature engineering on the dataset using Principal Component Analysis (PCA) to improve the accuracy of our models. This resulted in a large improvement in the performance of the XGBoost from 55.3% (Fig. 2) to 80.3% (Fig. 3) as seen in the figure below.

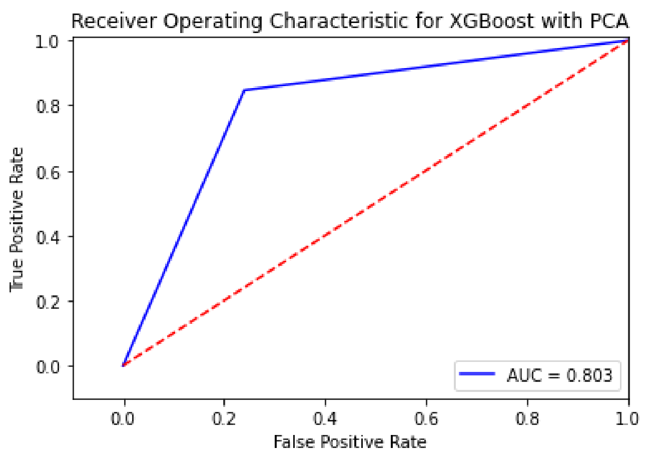


Fig 3: Preliminary results ROC curve for XGBoost with PCA

However, upon further analysis on the accuracies of the models, we found some fallacies in our implementation as we were only able to achieve a big improvement in performance for certain models (XGBoost & PCA), while other models remained the same.

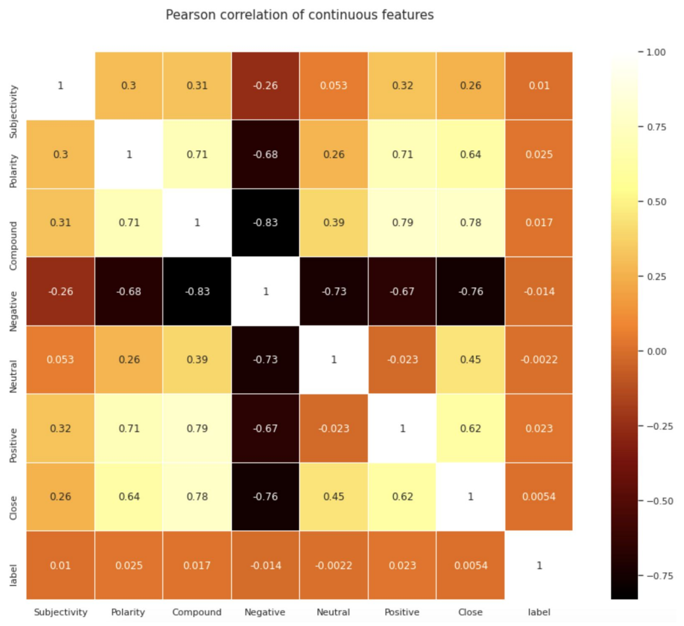
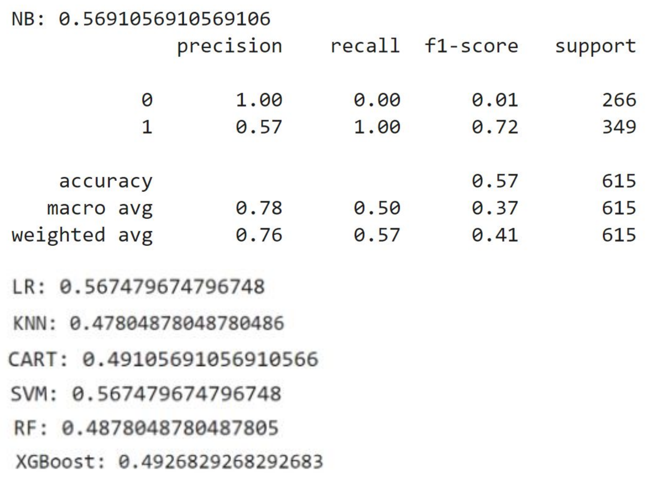


Fig 4: Pearson Correlation of Continuous features.

In Figure 4, we can observe that some variables such as Negative and Compound are negatively correlated. There are a few variables which seem to have a high correlation as well. We decided not to use any sort of dimensionality reduction technique to be applied since the number of dimensions we have is not considered high, and doing so would result in our model having lesser features to train on. We have removed any previously used dimensionality reduction techniques that we have mentioned during the poster presentation.

After taking another look at our dataset, we also found that the labels that we were using were generated were incorrect. The labels that we were using previously, were comparing the prices of the stock on the current day with the previous day, instead of comparing it to the next day. After cleaning up our models and dataset, we were successfully able to remove these discrepancies, and arrived with the following experimental results.



```
NB: 0.5691056910569106
              precision    recall  f1-score   support

           0       1.00      0.00      0.01       266
           1       0.57      1.00      0.72       349

    accuracy                           0.57       615
   macro avg       0.78      0.50      0.37       615
weighted avg       0.76      0.57      0.41       615

LR: 0.567479674796748
KNN: 0.47804878048780486
CART: 0.49105691056910566
SVM: 0.567479674796748
RF: 0.4878048780487805
XGBoost: 0.4926829268292683
```

Fig 5: Scores for Gaussian Naive Bayes and accuracy of other models

In Figure 5, we observed the respective scores for the Naive Bayes model. After training all our models again with the cleaned parameters, the accuracies for the models were more consistent with one another as seen above. With the new accuracies we retrieved, the model with the highest accuracy was neither LDA nor XGBoost but the Naive Bayes classifier model with an accuracy of 56.9%.

## Discussion

In this section we will be discussing some of the interesting highlighted research questions with their supporting evidence.

1. To answer the first 2 questions in our introduction, we can see from our preliminary experimental results that there is a slight correlation between news sentiments and the stock prices.

   From our experimental results, we can observe that by using the sentiment scores of the news headlines, our model was able to predict the next day shift of stock prices with an accuracy of around 57%, which is greater than the probability of randomly choosing if they increase or decrease (50%).

   From this, we deduced that there is a correlation between news with positive sentiments leading to an increase in stock prices as well as news with negative sentiments which led to a fall in the respective stock prices.

   While our model accuracy is not very high, our model does considerably well as compared to other models from Kaggle notebook with an average accuracy of 50% to 54%. We are still able to predict the shift in stock prices the following day with a slight amount of certainty. As such, for question 3, we can conclude that we are able to predict stock prices shift based on news sentiments with some form of certainty.

   However, taking a closer look at the relatively low accuracies, we deduce that there are still many other factors contributing to the shift of these prices that were not included. This can be seen in other research works to predict the shift in stock prices with a different input dataset. An example would be using historical stock prices to predict the next day adjusted close of the stock. Similarly, other factors that could contribute to improving the accuracy of our stock prediction model include market sentiments and tweets.

2. In Sentiment Analysis, the overall sentiment of a sentence is derived and used as features to train our model. As a result, the overall result of any classification using these sentiments will be affected by how distinct the selected features are.

   In our project, we split the sentiments based on the assumption that the Top News Headlines on a certain day will impact how the market moves on the next day. In order to make that assumption, we took a look into the headlines for words that could be an indication for the rise or fall of a stock.

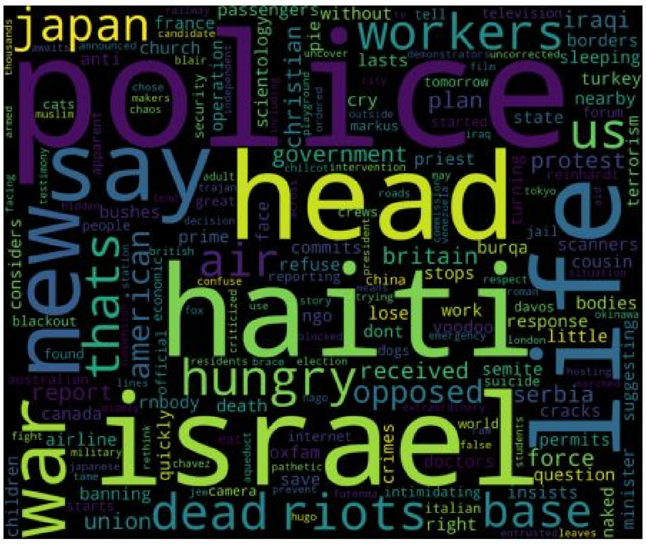


Fig 6: Decrease Words. Word Cloud displaying most common words found in New Headlines where stock prices decreased the following day.

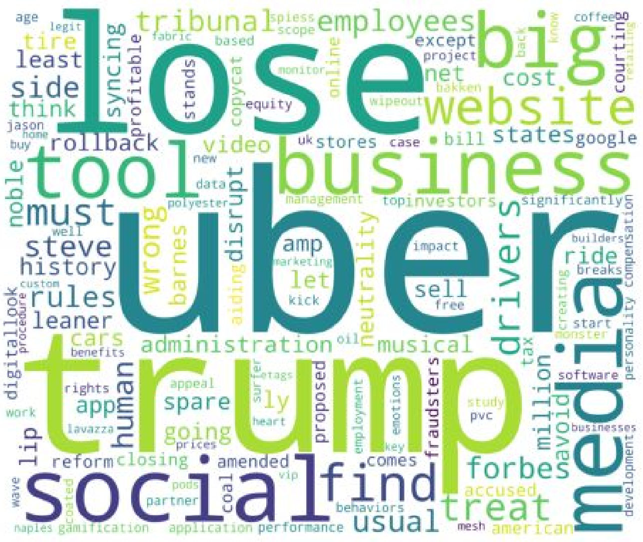


Fig 7: Non-Decrease Words. Word Cloud displaying most common words found in New Headlines where stock prices increased or stayed the same the following day.

By looking at the two word clouds, we found that most of the words are neutral words and they are mainly nouns which are not present in VADER lexicon and thus it does not hold any sentiment score. Therefore, the overall results of the classification may not be good since most of the news headlines that were used are generally neutral and have low positive or negative sentiment scores.

## Conclusion

Although we were able to get an accuracy of 81.4% and 80.3% using the LDA and XGBoost models with PCA respectively, we discovered some issues that led to the abnormally high accuracy compared to

the other models by analysing the steps we took to process the data.
When the data contamination was removed, we retrained our models and produced a set of results that were largely consistent with the Naive Bayes model achieving the highest accuracy.

However, in this project, we assumed that the market had perfect information and all news would have an effect on the stock trends. While media sentiment is an important factor, we should also consider other factors such as the past data of stock trends and the market sentiments as well. By incorporating these factors into our models, we will then be able to make a more informed and accurate prediction of the stock prices.

In addition, our current dataset uses reddit headlines to train our models. As our headlines were only taken from Reddit, it might not encapsulate the diverse set of headlines that are affecting the stock prices. Furthermore, most news headlines from Reddit are non-finance related which might not have an impact on the stock prices. As such, to extend our current project, we can make use of news headlines from media companies or even tweets to predict the shift in stock prices.

We would suggest a few future directions for this type of research. Firstly, as S&P 500 comprises around 500 stocks, it might be better to focus on a few individual prominent stocks and source for large news dataset centred around these stocks. Results could be better because by focusing on a fewer prominent stocks, the news dataset will have less noise and less variance during the prediction.
Secondly, social media comments on news dataset about the stocks could be used for sentiment analysis. Because psychological effect is a non-tangible factor that plays a crucial role in stocks prediction, sentiment of the tweets suggests the confidence level of the public on the stock. It may help to predict the stocks better.

Although our project did not consider other factors due to the time constraints, it forms a foundation where future work could be built on to improve the accuracy of stock predictions.

## Acknowledgements

1. We would like to acknowledge Prof Min and Prof Martin for their feedback during our project proposal. His feedback were insightful and we discussed as a team the changes to be made before moving forward to perform more analysis on our results.
2. We would also like to acknowledge our project TAs, Tongyao and Abhinav for identifying certain issues that our project has. We put some thought into them and made the changes respectively.

## References

[1] Bohmian. 2020. Sentiment Analysis Of Stocks From Financial News Using Python. [online]
[2] Devam, S. 2020. COMPARISON OF MODELS FOR FINANCIAL NEWS. [online]
[3] Jason Brownlee, 2017. Encode Text Data for Machine Learning with scikit-learn. [online]
[4] Kaggle Dataset. Daily News for stock market prediction. [online]
[5] Top 25 news article from Reddit from 2016 to 2020. [online]
[6] S&P500 SPY stock data, Yahoo Finance. [online]
[7] Hao Yee Chan, Anthony Chow. Using Newspaper Sentiments to Predict Stock Movements. [online]
[8] B. Johan, H. Mao and X. Zeng, " Twitter Mood Predicts The Stock Market", Journal of Computational Science, Vol.2, No.1, pp.1-8, 2011
[9] X. Li, H. Xie, Y. Song, S. Zhu, and Q. Li and F. L. Wang, "Does Summarization Help Stock Prediction? A News Impact Analysis", IEEE intelligent systems, Vol. 30, No. 3, pp. 26 – 34, 2015.
[10] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System". [online]