Title

* Introduction and Objectives

Talk about the amount of money and focus on analysing and predicting stock price movements.

Beyond financial fundamentals like profit margins and operating costs, there are alternative market forces which can drive change or volatility in a stock’s price. One which has arguably grown in prominence with increasing global interconnection through the internet, is public sentiment. People can now share their thoughts and opinions across extremely large audiences, influencing huge numbers of people who may go on to make trading decisions on the stock market, influencing stock prices. Emotions are not an insignificant factor in influencing a stock price, and can lead to rapid stock increases or devaluations, but cannot be accurately tracked by company fundamentals. This paper looks to review how sentiment analysis and machine learning techniques can provide insight into numerically translating market sentiment with a view to predict stock price movements.

Beneficiaries

Beneficiaries of this project includes anyone working within or involved in the field of stock market trading. This can include both large organisations like banks, and individual day-traders who are interested in understanding and predicting stock market movements. Furthermore, this research may be relevant to academics interested in the application of natural language processing (NLP) techniques to social media platforms like Twitter. The specific topic for this project is related to the company Apple but a number of the concepts discussed and utilised within this project may be used with a range of other applications involving NLP and Twitter, like business owners better understanding public sentiment towards their company or products in an unbiased situation outside of a survey or focus group. Furthermore, this research may be of interest to Apple itself and its employees. As a publicly traded company, the success of Apple from their shareholders’ perspectives may be viewed at a core, if harsh level, as their stock price, being directly related to the value of these shareholders’ holdings. This research may allow Apple as well as their shareholders to better understand the degree to which their stock price is related to, in reaction or precipitation of, public social media sentiment. It may also provide the stepping stone to understanding how Apple may influence or otherwise be a part of social media sentiment involving their products and company.

Success Metrics

This paper covers a strictly business problem, and therefore its overarching goal is to make profit by predicting the stock price movements as closely as possible. As this is primarily a regression problem, success can be measured by Root Mean Squared Error (RMSE), which measures the error between each true and its corresponding predicted value, which is then squared, the average taken, and the square root is taken of to ensure the result is ambivalent towards positive or negative. This will be evaluated on the holdout dataset which will be two months’ worth of data after the initial dataset (June and July 2021).

As the core issue is profit, a simple experiment will be performed on the optimal machine learning model to review whether an investment strategy following this model would return a profit.

* + Better understand the forces which drive the Apple stock price
  + Machine learning and sentiment analysis is able to quantifiably
* Context
  + Get some facts and figures about people doing this at banks

The total value of US-based publicly listed companies is extremely large at $47 trillion (Siblis Research, 2021). Apple forms a significant 5.12% at $2.41 trillion and is widely accepted as one of the most popular and influential companies in the world. Large corporations like investment banks and hedge funds spend large amounts of money, time, and resources to project how these stocks are likely to move based on a wide variety of metrics, market understanding, and news. As mentioned earlier in this project’s introduction, a medium which has grown exponentially in importance and usability over the last decade is social media sentiment analysis. As computers have become more powerful, machine learning on large scales have become more viable and social media is an ideal source of emotions, news, and market sentiment. If, for example, fear is building within the market, this may be visible and quantifiably measurable on platforms like Twitter. This project looks to build on other contemporary work in this field as discussed below to attempt to measure and predict Apple’s stock price.

The concept of psychology’s impact on stock prices is not new, with the basis of stock prices being undervalued or overvalued almost completely resultant from overly negative or positive mass psychology driving the price beyond its quantifiably logical price level. This paper will summarise the contextual review from the proposal below and add depth where appropriate.

There is strong literature in this field of utilising social media as a representation of sentiment to predict stock prices. Zhang, Fuehres, and Gloor found that average Twitter user sentiment in general could indicate how the Standard & Poor 500 (S&P 500) index would move as a whole (Zhang, et al., 2011). This paper provides an important foundation for a great deal of literature which comes after.

Meesad and Li effectively applied a sentiment analysis and an SVM linear algorithm to tweets on Twitter and focused Apple only, achieving a 90.34% accuracy for price movement (Meesad & Li, 2014). This is highly relevant to this paper, which has also elected to focus on Apple. However, it should be noted that the accuracy represented above is related to binary classification where the price either moves in a positive or negative direction. Upon reflection, as this paper is focusing on what is a business problem, there is some benefit to attempting to predict not only overall positive or negative changes, but also the degree of these changes, given that if an unusually high prediction is made, then the model should adjust accordingly and suggest a higher investment to maximise profit.

This paper was particularly motivated by Coyne, Madiraju, and Coelho’s paper on ‘*Forecasting Stock Prices using Social Media Analysis’* (Coyne, et al., 2017), which examined 17 stocks for a year’s worth of price movement data and utilised ‘twits’ from the social website ‘Stocktwits’ to create a prediction model. They were able to achieve an average accuracy of 65%, illustrating a degree of predictive power. An interesting relationship they captured was the optimal number of days between stock market sentiment and stock price movement. This is also something which this paper will review and reimplement.

Oh and Sheng have a similar paper on the stock predictive power of micro blog sentiment, where they support the notion that online social platforms, while at times irrational, can be used to predict stock market movements, with a weighted f-measure of 0.851 for one of their models (Oh & Sheng, 2011). A particularly interesting facet of this problem explored by Oh and Sheng is around the difference between ‘bullish’ and ‘bearish’ sentiment, where bullish refers to a positive sentiment and bearish refers to negative. A relatively accepted psychological phenomenon is that people tend to focus on and give more weight to negative information than positive, typically known as negative bias (Ducette & Soucar, 1974; Fisk, 1980). Translating this to stock sentiment context, this paper looks to explore also whether bearish sentiments are weighed more highly or have more significant relevance to stock price movements. Oh and Sheng also addressed a point earlier from Coyne, Madiraju, and Coelho regarding the number of days between social media sentiment and stock price movement, suggesting that investors may hold to their prior beliefs for a period of time when faced with new news, causing a delay in stock price movement (Daniel, et al., 1998; Chan, 2003). This paper will address this by using a similar strategy to Coyne, Madiraju, and Coelho and creating multiple models for a range of time periods between social media sentiment and stock price movement. This will ensure the optimal range is met, while also illustrating more of the relationship between new news and time delays to the stock market.

However, this paper also considers opposing work like Oliveira, Cortez, and Areal’s paper which similarly uses data from Stocktwits but disputes previous work by authors like Oh and Sheng, suggesting that sentiment is not a strong predictor for stock market returns (Oliveira, et al., 2013). Furthermore, they suggest Oh and Sheng did not utilise a large enough test period for significant results. However, this paper notes that Oliveira, Cortez, and Areal utilised a multiple regression model in this analysis, which this paper suggests may not be complex enough to fully capture the depth of sentiment and price movement relationship. This paper will look to match this concern by trialling multiple neural network-based models like Multilayer Perceptrons and Support Vector Machines, which will add scalability from model depth. Furthermore, this paper will utilise data from a test period of a year’s length to ensure sufficient length.

## Context Summary

Overall, one key point this paper will attempt is to treat this as also a regression problem as opposed to purely a classification as a number of previous papers in the field do. While classification does provide understanding on the overall relationship between social media sentiment and stock price movement and the degree of strength of such a relationship, but this paper believes a regression or a more developed classification approach with a higher number of classes will both provide a potentially superior method for maximising profit, as well as illustrate a greater depth of the relationship between sentiment and stock prices.

With regards to the frame of the business use case provided, this paper is open to trialling any appropriate machine learning model which may perform under the circumstances, and will experiment in particular with the Multilayer Perceptron, Multiple Regression, and Ridge Regression.

Given time, resource, and scope restrictions, this paper has elected to focus on the Apple stock only. This allowed the researcher to dedicate a greater deal of time and effort in better understanding and modelling the specific intricacies related to the Apple stock, providing a more in-depth analysis.

* + Quants?
* Methods
  + Data Gathering

Initially, this project looked to follow the work of Coyne, Madiraju, and Coelho by using text data from the trading social website Stocktwits (Coyne, et al., 2017). This site consists of essentially a trading-focused community where members may post their opinion into the public area and discuss. However, this paper found that since then, obtaining permission and access to Stocktwits’ database has become untenable through current company administrative limitations. Therefore, this paper reasoned that Twitter is a highly suitable alternative dataset. While it differs from the dataset used by Coyne, Madiraju, and Coelho, it is also likely to have a far larger base of information as Twitter’s monthly active userbase is approximately 310 million, compared to 1.5 million from Stocktwits (Roof, 2016; Statista, 2021). Furthermore, while Stocktwits is more likely to have an informed audience relative to the context of this project’s problem, this paper argues that more people invest in large blue-chip companies like Apple than is contained within Stocktwits, while also overall sentiment of the population and market is more likely to be better captured within a larger population like that of Twitter. Therefore, this paper concludes that Twitter is a highly appropriate source of sentiment data for the purposes of this analysis.

The primary dataset of text was sourced using the Twitter API to gather all tweets within the specified timeframe of 11/05/2020 to 11/05/2021 containing the hashtags “#apple”, or “#AAPL” (relating to Apple’s US stock market tag). Gathering this data was a significant activity and less straightforward than initially projected. The opening approach selected to obtain the tweets was the Postman tool. This has a user-friendly interface and an uncomplicated query builder. However, the researcher quickly noted that a core part of retrieving tweets from Twitter is the limitation on academic projects of a maximum of 500 tweets per request. While a key known as the “next token” provides a link to the subsequent part of the query result, it was clear the Postman tool was not easily optimised to automate gathering the full set of data. Therefore, this project reviewed alternative options to gather the necessary data. One option reviewed was Tweepy.

The alternative which was implemented was to adapt manual Python code which directly interacts with the Twitter API without a ready-made interface. This process was reasonably lengthy as there was a great deal of customisation required such as how to structure the loops, data formatting, and storage. Furthermore, the Twitter API has a number of restrictions on the number of tweets which may be requested over a variety of timeframes, and therefore a series of IF statements needed to be adapted to the code to pause it when necessary, ensuring the process would run uninterrupted and without any automatic throttling from the Twitter API. However, this process was positive through relative comprehensibility and flexibility as the dataset was constructed on the familiar and flexible Python Jupyter notebook interface. This meant it was relatively easy for researcher to adjust and optimise sections of the process to best fit this project’s aims.

The Apple stock price dataset was sourced directly from the NASDAQ website which contained a variety of information regarding the stock price, including its daily open and close prices, the trading volume, and the high and low for each day. For the purposes of this project, the daily price change was extrapolated and merged alongside the relevant dates for each text entry retrieved from Twitter, allowing the project’s models to view each tweet in context of that day’s price change.

## Data Pre-processing

Data pre-processing for textual data which this project is personally sourcing was a significant section of this project’s work, involving several steps and judgement considerations.

One significant pre-processing step was to ensure the full text of each tweet was delivered. While standard tweets were satisfactory, retweets had their text automatically truncated, minimising the available text data for these tweets. A retweet is “a re-posting of a Tweet. Twitter's Retweet feature helps you and others quickly share that Tweet with all of your followers” (Twitter, 2021).

This paper considered and concluded that while a retweet is not the same as someone writing and expressing their opinion, it can be one’s way of expressing one’s opinion through the medium of another’s words. Therefore, this paper can assume that if a tweet is retweeted, the user has read the initial tweet and feels a similar sentiment which they wish to echo amongst their online social group. Therefore, this paper has concluded that these retweets are as important as original tweets and will be treated as such. Consequently, the retweet text field truncation would need to be rectified to ensure the machine learning model had the most accurate picture possible of the Apple-related tweets on Twitter. This problem was ultimately solved by identifying which tweet the retweet referred to, and copying the original untruncated text from the original tweet onto the retweet, ensuring there would be no truncation and compromise with the sentiment analysis. Looping for this activity was briefly attempted with the “iterrows()” function but was found to take an unacceptably long time to process due to the large size of the dataset (3,387,880 rows). Therefore, this paper trialled several other methods and concluded that NumPy vectorisation was the optimal solution, at approximately 71,803 times faster than a standard loop and 224 times faster than the iterrows() function initially utilised (Droste, 2019). This was more difficult to implement than the simpler loop, but was accomplished by creating a second dataframe with only the retweeted objects which could be merged with the relevant original tweets to obtain the correct original text.

As mentioned earlier, the relevant price data was modified to illustrate the price change for each day and then merged against the tweet data to show the price change on the day for each tweet.

One initial obstacle this paper noted was that the price change data was only available for days on which the market was open, meaning that for weekends and public holidays, there would be no relevant price data. However, this paper quickly concluded that removing tweets that did not match would not make sense, as while sentiment may not, at the time, have an immediate impact on the price, it is logical that news and sentiment over this time may build up and impact the next day the markets do open, both through people independently planning to make trades on the day as well as implementing buy or sell orders to occur as soon as the markets open. Therefore, this paper created a function which would take the tweet date, assess whether it was a weekend or public holiday, and create a new date column with the next available date when the markets would be open. While functional, this paper notes that an assumption has been made with regards to the relationship between tweets made on non-trading days and how that impacts the price. This paper has assumed that the relationship does not change and simply accumulates till the next available trading window, whereas it is possible this does not capture the full extent of the relationship, specifically on certain public holidays which may promote unique behaviour. However, this paper has made the judgement that any missed particularities in this relationship are likely to be minimal in terms of their impact on the price change, and therefore has been covered as per the function created.

Cleaning the text data

The tweets collected by this project, while all referencing “#apple” or “#aapl”, will come with a huge variety of words, phrases, and noise which are not constructive for sentiment analysis. The overall goal of sentiment analysis is to convey the key words, phrases, and meaning numerically which can be inputted to a machine learning model. The analysis inputs need to be as simple as possible while condensing as much conceptual meaning as possible into the text inputs. Therefore, before any meaningful text analysis, the data must be cleaned. While text subjectivity makes this a difficult process to carry out with comprehensive coverage, there are particular processes and checks to remove or modify any common errant issues with the text which would otherwise compromise the sentiment analysis.

Special characters

Links

Many tweets encountered in this project contained webpage links to articles and sites. While this are relevant for a Twitter user to read to gain more information and otherwise better understand the tweet’s sentiment, as far as sentiment analysis goes, these links are ineffective in discerning meaning. One way in which this paper could have potentially allayed this problem would be to search each link and provide a sentiment analysis on the contents of each. However, this introduces an unregulated number of additional sources which this project has determined is beyond its scope to manage. Therefore, these links when found by identifying where words start with “http” were removed from the dataset.

Lowercase

Part of reducing the number of variations on words which are conceptually identical is ensuring all characters utilised in this data set are lowercase, further reducing unnecessary complexity to the TF-IDF process.

emoji

lemmatisation

One issue sentiment analysis may encounter is the significant variety of words available to describe conceptually similar ideas, or how affixes may alter the spelling of particular words, leading them to be read as separate entities by the machine learning model. Lemmatisation is the process of returning a word to its basic format as can be typically found in a dictionary. An example is changing “carried” to “carry”. This minimises the number of conceptually similar words being recorded as different objects and simplifies the machine learning model. It may also return stronger results as repetitions of the same concepts will now be represented more clearly and significantly, creating a more accurate model. This project utilised the Natural Language Tool Kit (NLTK) library in a small function to firstly tokenise the tweet strings, which separates each tweet into its individual words which is a requirement for lemmatisation. This allows the researcher to consider each word as an individual element in isolation, allowing for effective lemmatisation. Then a lambda function was utilised to efficiently lemmatise each word, which was then detokenised to recreate the tweets. This last stage is important to ensure the TF-IDF function and resultant model can clearly understand each tweet’s word structure and process them in a computationally efficient manner.

Stopwords

Stopwords are common words like “the” or “is” which do not hold any conceptual meaning but are extremely common. Removing these are important to ensure these do not impair the sentiment analysis by being considered as features. These words were removed in this project by using the NLTK’s list of English stopwords, comparing against this and removing where appropriate.

* + Data Analysis
  + Requirements Analysis

Given the basis of this project is a fundamental business problem, the aim and subsequent requirements are clear. The overall goal is to maximise profit which can be gained by predicting the stock price movements as closely as possible. Since this is a regression problem, metrics like accuracy are not relevant, so success will be measured primarily through Root Mean Squared Error (RMSE). Furthermore, this paper will also explore the subsequent two months (June and July 2021) after the initial data collection as a holdout set to test the data on and collect final results.

* + Design

Following the requirements, as this is strictly a business problem and not a pure academic exercise, this paper has imposed no restrictions on the models utilised and reviewed a wide variety to better understand which performs the best with the given dataset. An array of regression models have been trialled, as well as several classification models, achieved by binning the data. This paper’s motivation for this is because there is no specific reason why this needs to be a regression problem, and indeed a classification approach may provide higher performance through providing an alternate data pattern. Moreover, it could be said that the investment decision is, to an extent, suited to classification as there are only three main options, buy, sell, or no action. However, this paper believes it is fair to say there is scale to these options, with a highly positive buy suggesting a larger investment, and therefore will review both regression and classification.

MLP

The Multilayer Perceptron (MLP) is a supervised learning feed-forward network with three layer categories, input, output, and hidden. Random weights are assigned to each neuron in the hidden layer to ensure they do not get stuck in local minima, influencing an output as values are passed through the system. Backpropagation then occurs when the output value is different to the true value, and the hidden layer’s weights are correspondingly updated to better fit the data. MLP is a popular tool for both classification and regression problems as it is accurate, scalable, relatively quick to train, and highly customisable.

Icon

Description automatically generated

Figure 1: A Concept of a Multilayer Perceptron

MLP Hyperparameters

This paper has discussed below the MLP hyperparameters which are available for optimisation.

Neuron Number

When discussing neuron number, there are three different relevant categories, the input layer, the hidden layer(s), and the output layer. The neuron number for the input and output layers are fixed to the dimensions of the dataset and desired output. The input neuron number will be fixed to the number of total words in the corpus after the TF-IDF function, which will be discussed further below. To summarise for this section, each unique word in the dataset will be considered with a separate input neuron which is fixed and cannot be optimised as part of this. Similarly, the output neuron number is fixed as there is only one desired output, an overall prediction for the stock price. The hidden layer however can be optimised to improve performance. Too few neurons may mean there is not enough capacity within the model to fully learn the dataset provided, leading to underfitting. Alternatively, too many neurons will likely lead to overfitting as the model adapts to noise unrepresentative of the population, as well as increasing computational time and effort. Heaton suggests the value could be 2/3’s of the input layer, plus the output layer, or also less than 2 times the input layer (Heaton, 2015). WRITE MORE PLS

Learning Rate

Learning rate refers to the rate at which the model adapts to the results of each epoch. In each epoch, the model will review which of its predictions were incorrect and the error between each prediction and its corresponding true value. The model will then adapt to this, attempting to reduce the loss. Learning rate is the rate at which the model adapts. Increasing this may lead to a faster training model or a model which is overall superior through more training, but may also overtrain or train past what would have been optimal solutions, without allowing enough epochs between them to clearly examine them.

Momentum

Momentum is closely related to learning rate and refers to an additional factor added to the neuron weighting calculations to increase training speed and ensure the model does not get stuck in local minima. What this refers to is how when training, a model may reach what appears to be locally the minimum loss for the model. Without momentum, the model may remain in this local minimum, despite the possibility that the global minimum may be beyond. Momentum allows the model to move past local minima to more fully explore and find the global minimum. There is research however which does suggest that local minima are relevant and at times, spending additional effort to find the global minima may also result in an overtrained model (Choromanska, et al., 2015). Taking this into account, this paper will opt for a balanced approach, considering a range of momentum values.

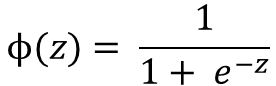
Epoch Number

Epoch number represents the number of times the model retrains with the aim of iteratively improving each time. In this regression problem, this is measured by training and validation loss. Increasing the number of epochs will likely improve model performance but may also cause overfitting. This occurs when a model is fit too well to a training dataset and has also modelled what is meaningless or counter-productive noise within that specific dataset. Therefore, when this model is applied to a test or holdout set, performance decreases as said noise is not present in the population, represented by the test set. To better manage this, this paper has elected to use early stopping as a form of regularisation. This observes validation loss and concludes model training when this metric has stopped improving in a meaningful way for a particular number of epochs known as the patience value. This is set at 5 epochs.

Activation Function

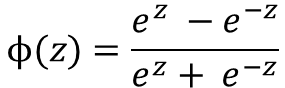
The activation function is a transformation applied to each hidden layer which supports neuron weight development. This is a key step as it is a significant part of how an MLP can be more sophisticated than a simple linear regression model. There are three main activation functions available to evaluate, rectified linear activation function (ReLU), Sigmoid, and Tanh. ReLU is a common modern activation function which does not suffer from the vanishing gradients problem like other neural networks. This problem occurs when a deep neural network (one with many layers) cannot properly complete backpropagation, meaning useful learnt information from the end of the model does not return to the beginning of the model. ReLU can be described as returning the input, or 0 if the input is 0 or less. Despite this initially seeming potentially very inconvenient for datasets containing a large number of negative values, weights are applied to each neuron, transforming them so they are likely still of use to this project’s dataset. Furthermore, literature has shown that ReLU trains up to six times faster than Tanh, another alternate activation function (Krizhevsky, et al., 2012).

The sigmoid activation function simply takes the input value and returns a value in the range of 0 to 1, with more positive values returning an output closer to 1 and more negative returning closer to 0. The function which governs this relationship is:

 (Sharma, 2017)

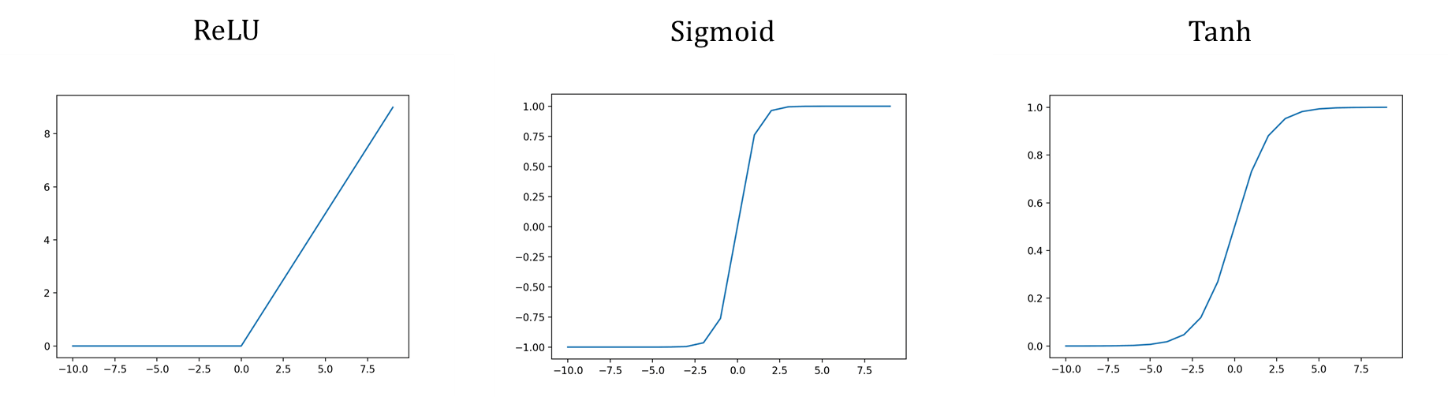
Where: z = the input value

The tanh activation function is similar to sigmoid but its lower bound is -1 as opposed to sigmoid’s 0. This is calculated as:



Where: z = the input value

These activation functions are graphically represented as below:



(Brownlee, 2021)

The above activation functions will be trialled with this project’s MLP to assess which is most suitable. Alongside performance with regards to RMSE, computational time and effort will also be assessed as this project is managing a significant level of data. Furthermore, the business application for this program may require quick retraining with new data available at a near-constant flow through new tweets, thereby reinforcing the need for a model which is computationally efficient.

Hidden Layer Number

The number of hidden layers is also a hyperparameter which may be optimised for this project’s MLP model. Increasing the number of hidden layers will increase the depth of the model, which may lead to greater development of the neuron weights, leading to a more accurate model. However, too many hidden layers may lead to the vanishing gradients problem, particularly when not utilising the ReLU activation function. Furthermore, the model may overfit to the training dataset, reducing test performance while also requiring unnecessary computational effort.

Multiple Regression

Multiple regression is a form of linear regression which attempts to fit a linear model to the dataset which minimises the residual sum of squares between the true values and predicted. The term ‘multiple’ refers to how it considers multiple variables to model this relationship. In the case of this dataset, variables refer to the total number of words used in the dataset, also known as the corpus, where each unique word is represented by a variable. This is a simpler model when compared to MLP, but this project has determined it is worth investigation, particularly as Oliveira, Cortez, and Areal also utilised this model, which allows this project to more accurately compare results and review their conclusions.

Optimisation

Ridge Regression

Ridge regression is similar to multiple regression but uses L2 regularisation to minimise any issues relating to multicollinearity which may arise from using multiple regression. This is performed by imposing a penalty on coefficient size equalled to the magnitude of coefficients, squared. This reduces the impact any collinearity may have on the results. Multicollinearity was expected by the nature of sentiment analysis where some words have conceptually similar meaning, and therefore it stands to reason that their ratings may be similar. However, this project has taken steps to mitigate this through lemmatisation as mentioned earlier, which reduces words to their core meaning, reducing any potential collinearity. However, it is possible that not all the words within the corpus were correctly identified and lemmatised, and therefore ridge regression is an acceptable model to trial for this project.

Optimising Ridge regression is also a relatively simple matter as there is only one meaningful hyperparameter available for optimisation, the alpha value. This refers to the regularisation strength, which is the penalty applied to the variables to reduce collinearity. This was optimised by iteratively searching a logarithmically spaced range between 0.00001 and 100 and scoring by RMSE, where the optimum alpha value was found to be 2.68.

Lasso

Lasso regression is also similar to multiple regression but uses L1 regularisation, which applies a penalty to features which are determined to be less important as a result of the quantity of zero values they contain. Therefore, a rare word which does not appear much within the dataset is more likely to be ignored by the model to reduce complexity and improve performance. This model was selected this project has an especially large number of words/features (5844) in a sparse model with many zeros, which suits the usage of Lasso (Nagpal, 2017). This model has the same hyperparameter as ridge regression and therefore has been optimised in the same way, using a logarithmic scale to iteratively test alpha values. PUT ALPHA RESULT HERE

Elastic Net

Elastic Net regression combines L1 regularisation from Lasso and L2 regularisation from Ridge. Given the reasoning above for including Lasso and Ridge, it is pertinent to include Elastic Net within the models trialled by this project. Hyperparameters for this include alpha as discussed earlier, and also l1 ratio, which is the ratio of L1 to L2 regularisation this model utilised, with a high value emphasising L1, and a lower value emphasising L2. PUT RESULTS HERE

* TF-IDF

TF-IDF summary

Sentiment analysis is essentially transforming qualitative text into numerically tangible data which can be read and analysed by a machine learning model. This is done by analysing each word given the context and providing it with a weight which denotes its importance. There are several different methods by which this sentiment analysis may be performed, with this paper reviewing three methods of word weighting term frequency (TF), inverse document frequency (IDF), and term frequency – inverse document frequency (TF-IDF).

TF utilises the frequency of terms used within the text to identify importance but lacks as it does not consider these terms with regards to the corpus as a whole, and therefore lacks perspective. IDF considers a word’s importance as the total document number divided by document number including the particular word. This is an advantage over TF since it considers the entire corpus as opposed to a single document which is unlikely to be 100% representative of the population, given the variety of tweets and language people may use.

TF-IDF considers the importance of a word relative to the document and the dataset corpus. Initially proposed in 1972 under the phrase “term specificity” (Jones, 1972, p. 11), it has been shown to be more accurate than TF (Cui, et al., 2015), likely because it considers the entire corpus as opposed to a single document, leading to a more grounded feature score. It was also found to be significantly more accurate than IDF as well, leading this paper to consider TF-IDF as its primary tool for sentiment analysis, combining benefits from TF and IDF.

The TF-IDF function from Sklearn returns a Scipy sparse matrix, which is a high-level representation of each word contained within the tweet dataset and its importance weighting. This by itself is not immediately useful to the project, but can be utilised to create an array representation which quantifies each tweet and enumerates each word in the corpus, numerically representing each tweet in the context of all the words utilised throughout the dataset. This can then be used as an input into machine learning models which will then attempt to predict the price change relating to each tweet.

Language

Twitter is a global company, and therefore has users from across the world, speaking and using different languages with different scripts. Each language will have unique phrases and meanings which are not shared by others, which may pose some difficulty for a machine learning model. As people over the world may invest and otherwise influence Apple’s stock price, it is relevant to attempt to examine these differences and this project will attempt to create separate models by language to review its impact on performance.

* + Implementation
* Randomsearch vs Gridsearch
  + Say why randomsearch is better for this

For the MLP hyperparameter optimisation, there were multiple options available. The two most optimal were found to be Gridsearch and Randomsearch, both functions from Sklearn. Explain what they are

While Gridsearch is generally more thorough, it is extremely time intensive as it requires the model to iterate for each possible hyperparameter configuration. Contrary to this, Randomsearch does not run every configuration, but instead runs a number of randomised configurations according to the researcher’s prior set hyperparameter distribution. This has been found to be far less time and computationally intensive, as well as producing strong results.

For hyperparameters where Randomsearch was not appropriate, a Gridsearch-style iterative approach was taken to iterate through potential options like activation function, neuron number, and hidden layer number.

* + Testing/Evaluation
* Experiential Results
* Discussion and Critical Evaluation of Findings
* Evaluation, Reflections, and Conclusions
  + Lessons Learnt
  + Future Research
* References