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, operating costs, and more, 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.

* + 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
  + 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.

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 …

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 truncation of retweets was rectified 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.

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. 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. 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.

* + Data Analysis
  + Requirements Analysis
  + Design

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, 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.

* TF-IDF

TF-IDF summary

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 its 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

* + 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.

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