6CCS3PRJ Final Year

Analysing the relationship between stock prices and social media sentiments

Final Project Report

Author: William Cooper

Supervisor: Professor Peter McBurney

Student ID: 20046717

April 5, 2023

Abstract

This project aims analyse the relationship between the sentiment on social media towards brands and companies and the share price of same. Specifically, the project looks to use sentiment analysis techniques to identify the attitude towards a company over long periods of time and look for patterns relating to increases or decreases in the value of the company.

The overall goal of the project is to create a functional piece of software that creates an accurate sentiment analysis model which can then correctly identify the sentiment of a large number of social media posts regarding different companies, combine this data with historical share prices and then generate readable graphs that can be used to identify patterns and ideally predict future share prices based on current sentiments. This code should be easy to use and alter so that it can be used over any time span with any company with a level of detail that is up to the user.

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 25,000 words.

William Cooper

April 5, 2023

Acknowledgements

I would like to thank Professor Peter McBurney for overseeing and supporting me throughout this project, your guidance was invaluable. Also thanks to my family for all the support.

Contents

1 Introduction **3**
- 1.1 Project Aims . . . 3
- 1.2 Report Structure . . . 4

2 Background **5**
- 2.1 Sentiment Analysis . . . 5
- 2.2 Share Price . . . 6
- 2.3 Twitter . . . 7
- 2.4 Natural Language Processing . . . 7

3 Requirement & Specification **9**
- 3.1 Requirements . . . 9
- 3.2 Specifications . . . 11

4 Design **15**
- 4.1 Software Behaviour Flowchart . . . 15
- 4.2 Sentiment Analysis Model . . . 16
- 4.3 Modular Structure . . . 16
- 4.4 Third-Party Content . . . 19

5 Implementation **21**
- 5.1 Version 1 - Initial Sentiment Analysis Model . . . 21
- 5.2 Version 2 - Improved Prediction Accuracy and Expanded Range of Stocks . . . 23
- 5.3 Version 3 - Increased Number of News Sources and Graph Clarification . . . 24
- 5.4 Testing . . . 25

6 Legal, Social, Ethical and Professional Issues **30**

7 Analysis **31**

8 Evaluation **40**
- 8.1 Sentiment Analysis . . . 40
- 8.2 Graph Generation . . . 40
- 8.3 Application Speed . . . 41

9 Conclusion and Future Work **42**
9.1 Conclusion 42
9.2 Future Work 43

References **46**

A Extra Information **47**
A.1 Final Generated Graphs 47

B User Guide **58**
B.1 Instructions 58

C Source Code **60**

Chapter 1

Introduction

Sentiment Analysis involves of computationally categorising the attitude or opinion of a piece of text, often using machine learning and classification algorithms, and is frequently employed to survey the views of large groups of people towards an entity or topic. It is commonly used by companies to assess the attitude of potential consumers towards their product, as well as evaluate the popularity of various political figures and important issues, reactions to decisions and other polarising ideas that may lead to many different viewpoints from large numbers of people online.

1.1 Project Aims

The primary aim of this project is to find any patterns or correlations between the sentiment towards a company online and that company's stock price at the time, to see the degree to which social media can affect the value and perceived value of a company, as well as look at the viability of using sentiment analysis of social media posts to predict future stock fluctuations and make more profitable investing easier. The project itself consists of three main parts. First, the sentiment analysis model, which is created and trained using existing publicly available datasets. Second, using that same model, now trained, to analyse the sentiments of a massive set of social media posts related to a selected list of companies. Third, plotting graphs showing the stock prices and sentiments of each company over time in an attempt to show useful and interesting patterns and correlations.

1.2 Report Structure

This report begins with a background overview of the key concepts that are relevant to the project, going into enough detail such that the scope of the project is understandable. This is followed by the specification of the requirements of the project, it's design and an overview of the various stages of its implementation. After is a brief summary of the legal, social, ethical and professional issues relevant to the project, analysis of the results shown, evaluation of the entire project and the final product and a summarising what has been learnt and how the project could be improved and continued in the future.

Chapter 2

Background

2.1 Sentiment Analysis

Sentiment Analysis is the process of categorising the attitude or opinion of a piece of text using a variety of techniques, including Natural Language Processing (NLP) and different machine learning classification algorithms, such as Logistic Regression and Naive Bayes.

It is used to take large amounts of plain text statements and automatically ascertain the overall sentiment of the text, often with respect to a particular keyword or subject. Sentiment Analysis is commonly used by companies and brands to help them stay aware of their own reputation, as well as to be able to see how specific decisions that have been made can affect the sentiment of customers towards them. It can be used to see how different trends can affect the way in which customers see different products and services, and also be able to acquire a ”gut reaction” so to speak and understand the first instincts that consumers have towards specific ideas or products, particularly if the data being analysed is limited in size and scope, such as individual social media posts or headlines.

2.1.1 Existing Published Work Related To Sentiment Analysis of Social Media

Sentiment Analysis of News Tweets

[1] This work is an example of the sue of sentiment analysis and its applications for social media. It is centred around using sentiment analysis to classify social media posts by major news outlets related to a specific location, in this case Dubai. At its core, it is similar to my project in that it uses a relatively simple and easy-to-implement method for sentiment analysis,

using Python and the Natural Language ToolKit[2] to classify tweets. While this existing work was more concerned with researching the accuracy of sentiment analysis and identifying the most common keywords that identify positive or negative sentiments, I chose to extend sentiment analysis in a different direction, focusing more on identifying patterns and trends that could be used to predict future share price increases or declines.

Social Media Sentiment Analysis based on COVID-19

[3]This project was concerned specifically with sentiment analysis of tweets related to COVID-19. The use of two different sentiment analysis methods, a Recurrent Neural Network and a third party sentiment analyser, in this case TextBlob. This project had one particular aspect that I considered extending, which was the use of degrees of sentiment. Rather than just classifying tweets as positive or negative, this project included strong and weak variations of positive and negative sentiments, as well as a neutral sentiment This allowed for much more accurate results showing the range of different emotions, as opposed by the limited choice of two sentiments which are not always applicable, particularly to financial news. However the increased complexity this would bring, especially when considering that identifying sentiment accurately isn't the be all and end all of the project, rather an important step needed to analyse the financial data that is obtained, meant that this was not viable.

2.2 Share Price

Share Price refers to the cost to buy a single share in a company. The share price of a company is not fixed and changes based on demand and the number of shares available on the market. It can be affected by innumerable factors, such as changes in management, inflation, over-performing or under-performing according to forecasts, public sentiment towards the company, or changing trends.The share price of a company is constantly changing and will rise and fall constantly every day that the markets are open. [4]

Share price is a useful metric for tracking the growth or decline of a company over any length of time, and it requires the use of long term historical data in order to model the changes accurately and get a clear understanding of how it has changed over time. This data can be found in a variety of databases and financial websites, such as Yahoo Finance.

2.3 Twitter

Twitter is a social media platform and microblogging service, founded in 2006, that allows users to create accounts and post messages called "tweets". These Tweets are able to be seen by anyone and were originally limited to being 140 characters in length, in order for them to be compatible with SMS character limits, although as of 2017 the length limit was increased to 280 characters.

While Twitter was founded in 2006, it was between 2008 and 2010 that the service began to grow in popularity, with the average number of daily tweets increasing from 300,000 in 2008 to 35 million in 2010. It was during this same period that many traditional media sources and publications began to create accounts on the platform to share news, particularly headlines with links to complete articles located elsewhere. It is these news accounts that tend to produce the most tweets [5], due to the fact that they often create tweets automatically to share news articles, or use third-party web applications to allow them to manage their social media presence more effectively.

Twitter is one of the social media platforms best suited to the spread and reception of news. This is due in large part to the ability of users to "retweet" posts made by other users. By default tweets are shown to users in reverse chronological order, with each users timeline consisting of tweets that come from accounts that they already follow. The retweet functionality allows tweets to be seen by users that don't follow the account that tweeted it, allowing any tweet, but news related tweets in particular, to be seen and shared by many people very quickly, which is a big part of the environment that leads virality on social media and allows Twitter to function so perfectly as a tool for learning about and sharing news.

2.4 Natural Language Processing

Natural language Processing is defined as "a form of computational linguistics in which natural language texts are processed by a computer (for automatic machine translation, literary text analysis, etc.)" [6]It is a broad field of both computer science and linguistics concerned with making natural language data, which is often full of complicated subtleties and nuances and processing it such that it is able to be understood and analysed by a computer system.

It almost always involves tokenizing data, which is the process of breaking up natural language into smaller parts called tokens, which can consist of whatever the user needs, including single words, URLs, emoticons, or individual characters. This is followed by the process of

normalisation, where words are converted into their canonical forms, for example, converting ”studying” into ”study” or ”multiplication” into ”multiple”. Two common methods of normalisation are Stemming, which removes suffixes to leave us with stem words, and Lemmatization, which uses knowledge of word structure and vocabulary to convert words into their roots.

Natural Language Processing has many different uses, due in large part to the massive difference between the languages humans use to communicate with each other and the way humans can communicate with machines. Some of the common uses of Natural Language Processing include automatic translation services, such as Google Translate. When translating individual words, a dictionary of terms is generally sufficient however, when attempting to translate entire sentences from one language to another, the positioning of words and their meaning within a wider context can lead to massive discrepancies, and as such NLP is extremely important. Chatbots and virtual assistants are another area of computing that relies heavily on Natural Language Processing, as the ability of these tools to discern the meaning of questions as a whole that are posed to them by users, as opposed to understanding individual words or tokens.

Chapter 3

Requirement & Specification

The primary goal of this project is to try to find a concrete relationship between the overall social media sentiment towards a publicly traded company, and the way in which that stock performs in the future. Specifically, the way in which the historical financial data of a company correlates to the overall sentiment on Twitter at the same time, which is to be understood by analysing all the tweets from a selection of verified financial news accounts and calculating the strength and positivity or negativity of the sentiment at each interval.

3.1 Requirements

In order to complete the project, there are four overall requirements that need to be completed: The text and date of Tweets from verified financial news Twitter accounts need to be gathered for a number of companies, the historical share prices for stocks need to be obtained, the code that is required to organise all of the financial data into intervals and analyse the sentiments of the content of the Tweets needs to be completed, and the graphical representations of the results need to be generated.

3.1.1 Twitter Data

- Twitter data should come only from verified financial news accounts.
- Twitter data consist only of tweets that contain the name of the company being analysed.
- Twitter data should consist of the text of the Tweet and the date it was posted.
- Twitter data should deliberately exclude replies and retweets.

3.1.2 Financial Data

- The financial data should consist of the relevant share price information, such as opening and closing prices, high and low points, and the date from which the data was taken in a day / month / year format.
- The financial data should contain the relevant prices for the stock at weekly intervals from the maximum time frame or since 2013, whichever is shorter.

3.1.3 Code

- The code should iterate through the test dataset and split each tweet into a list of individual words (tokenization), and separate the test tweets according to whether they have positive or negative sentiments.
- The code should remove special characters, punctuation, and links from the list of tokens.
- The code should convert each token to it's normal form.
- The code should create, train, and test a sentiment analysis model using this tagged dataset.
- The code should iterate through each week from 1st January 2013 onwards, and analyse the sentiment of all scraped tweets from that week, incrementing counters that track the total number of tweets, total number of positive tweets analysed and total number of negative tweets analysed for that week.
- The code should create a dictionary, with each key a week the tweets came from, and each value being the net sentiment of the analysed tweets for that week.
- The code should generate graphical representations of the data that has been collected.

3.1.4 Graphical Representations

- The graphical representations that are generated should consist of two separate graphs. One should be a line graph showing the stock price each week, and the other should be a bar chart with each bar showing the net sentiment of all the finance tweets that were analysed that were posted that week.

3.2 Specifications

Each row in the table consists of the requirement itself, a specification detailing how this may be achieved, and the importance of completing each requirement with regards to the wider project.

Requirement Details	Specification	Importance
The Twitter data should only come from verified financial news accounts.	Use the SNScrape and TwitterSearchScraper module to write a query that only scrapes accounts from a preset list.	High
The Twitter data consist only of tweets that contain the name of the company being analysed.	Include name of company being analysed as a parameter in SNScrape query.	High
The Twitter data should consist of the text of the Tweet and the date it was posted.	Include tweet content and tweet date as parameters to be saved to CSV.	High
The Twitter data should deliberately exclude replies and retweets.	Check if tweet content contains "@" or "RT" tags, and exclude if they are found.	High
The financial data should contain the relevant prices for the stock at weekly intervals from either the maximum time frame or since 2013, whichever is shorter.	Query Yahoo Finance to search for historical data and set interval to weekly.	High

The financial data should consist of relevant information about the price of the shares, such as opening and closing prices, high and low points, and the date from which the data was taken in day / month / year format.	Download historical data from Yahoo Finance as a CSV file.	High
The code should iterate through the test dataset and divide each tweet into a list of individual words (tokenization), and separate the test tweets according to whether they have positive or negative sentiments.	Iterate across dataset and use the sentiment string ”Positive” or ”Negative” to divide the tweets, and use the .split method to tokenize the tweets into individual words.	High
The code should remove special characters, punctuation and links from the list of tokens.	Include function to replace any tokens consisting of special characters with empty strings, and use regex matching to replace and URLs with empty strings.	High
The code should convert each token to it's normal form.	Use Lemmatizer from NLTK library to convert words to their normal form, and check outcome not in the stopwords list (”a”, ”and”, ”the” etc.)	High

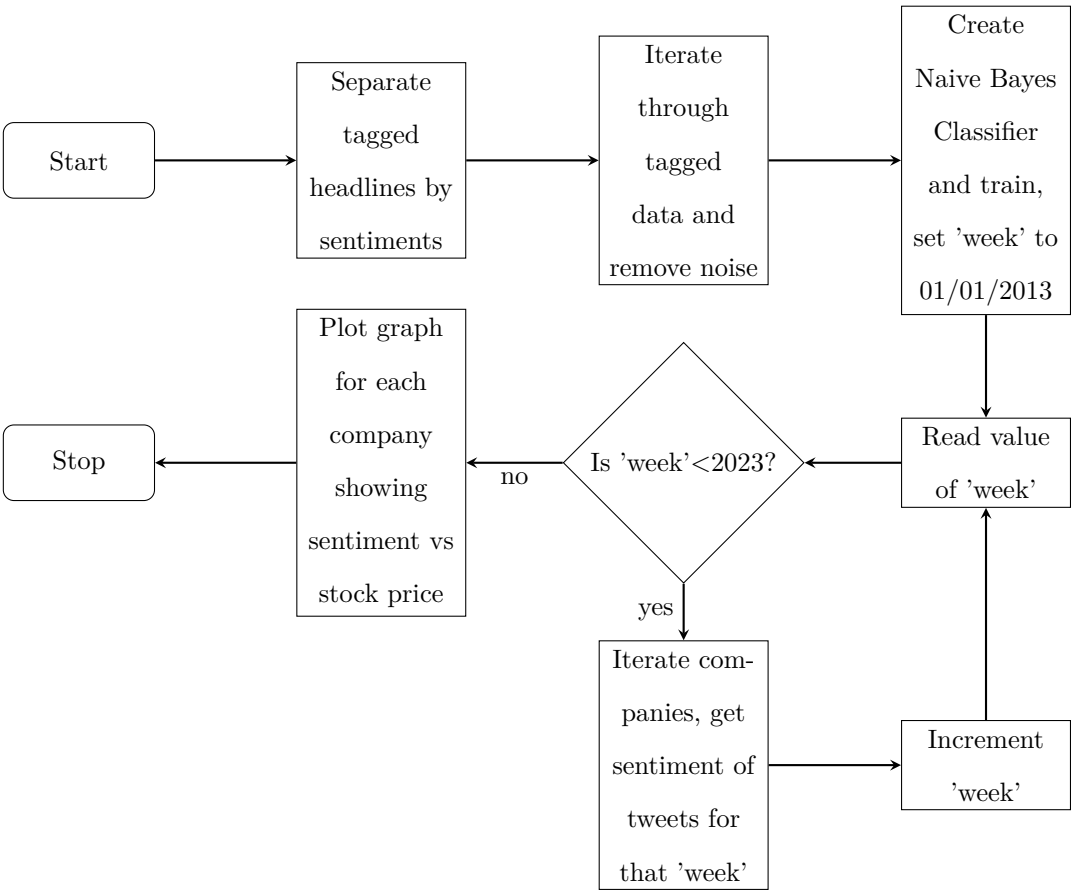
The code should create, train, and test a sentiment analysis model using this tagged dataset.	Create combined dataset of cleaned tokens tagged with "Positive" or "Negative" sentiments as appropriate, randomise the order of this dataset, import Naive Bayes classifier from NLTK library and divide data 80:20 for train and testing data.	High
The code should iterate through each week from January 1st 2013 onwards, and analyse the sentiment of all scraped tweets from that week.	Incrementing the date by a week each time, loop through the scraped tweets, and analyse the sentiments. Increment counters that track the total number of tweets, total number of positive tweets analysed and total number of negative tweets analysed for that week.	High
The code should create a dictionary, with each key being the week the tweets came from, and each value being the net sentiment of the analysed tweets for that week.	After the value of the week is incremented, add the current week and net sentiment value to the dictionary as a key value pair and reset the counters i.e. ["01-01-2013": 75]	High
The code should generate graphical representations of the data that has been gathered.	Use the Matplotlib library to generate graphs using the CSV files and dictionary's generated after the sentiment analysis.	High

The graphical representations should contain two separate graphs. One should be a line graph showing the stock price each week, and the other should be a bar chart with each bar showing the net sentiment.	Use Matplotlib to plot a line graph to show the stock price changing over time, then overlay a bar chart to show the net sentiment for that week, to clearly see how net sentiment relates to stock price changes.	Medium(as it may be the case that only one graph is needed, or the graphs that are generated are of different types than are outlined here).

Chapter 4

Design

4.1 Software Behaviour Flowchart



4.2 Sentiment Analysis Model

The sentiment analysis model chosen was Naive Bayes, as implemented by Natural Language Toolkit library. [2] The Naive Bayes classifier works by taking a set of training data, each item tagged with it's classification, in this case "Positive" or "Negative". Each token within the training data has a probability calculated for it occurring in either "Positive" or "Negative" headlines based on the number of times it appears and the total number of headlines for that classification.

With a multinomial event model, samples represent the frequencies with which certain events have been generated by a multinomial $(p_1, \ldots, p_n)$ where p_i is the probability that event i occurs. In the case of this project, the event that occurs is the presence of the token i that appears in a headline. $\mathbf{x} = (x_1, \ldots, x_n)$ is then a histogram, with x_i counting the number of times event i was observed in a particular instance. The likelihood of observing a histogram x is given by

$$p(\mathbf{x} \mid C_k) = \frac{(\sum_{i=1}^{n} x_i)!}{\prod_{i=1}^{n} x_i!} \prod_{i=1}^{n} {p_{ki}}^{x_i}$$

It then uses these individual token probabilities and the ratio of "Positive":"Negative" headlines in the dataset to calculate a score for each headline and for each sentiment, with the higher scoring sentiment being the model's prediction of the classification of the Tweet.

4.3 Modular Structure

The system itself is could in theory be designed as a single function or script, as it takes no input from the user and has no variance in terms of what it generates from one run to another, although the output may change slightly as each time the sentiment analysis model is trained and tested it may result in subtle and generally unnoticeable differences in it's accuracy as well as how it categorises certain inputs when it is used on real tweets from media companies later on. However, in the interest of making it easier to understand, to test, to make changes and to actually use the software it was designed such that the code consists of 6 different functions, as well as the main script of the code that is run whenever the file is opened with the python3 command from the command line.

4.3.1 Twitter Scrape

The twitter_scrape function is the only function within the code that is not automatically called when the script is run from the command line. There are two main reasons for this. The first is that if the function has been run previously and the files that it generates are still saved in the same location then running it again is completely redundant as all it would do is replace those existing files with copies of them. The second reason is that running the function is time consuming, and dependent on not only the processing power of the computer on which it is being run, but also the speed and reliability of its internet connection.

The twitter_scrape function is used to create the 10 csv files that contain all the tweets from the list of Twitter accounts found within that reference any of the stocks that are to be analysed. The functions iterates across a list of stock companies, and searches for tweets mentioning that stock from each of the 16 Twitter accounts, adding the date and text of those found to the corresponding csv file before moving on to the next stock. This means that millions and millions of individual tweets are being checked to see if they contain a single word, saving them to a list, then converting that list to a dataframe and then a csv file.

Having this function within the main body of the code would ensure that those tweets are always accessible to the software, and would guarantee that the code would run correctly every time, however the amount of time it takes would make the software extremely inefficient as the volume of tweets that have to be analysed is so high that the rest of the program is already much slower than would be ideal, without wasting time regenerating resources that have more often than not already been created.

4.3.2 Remove Noise

The remove_noise function is used to convert the words contained within the training dataset into clean tokens that can be used to analyse the sentiments of future tweets. It is kept as its own separate function or module because it is called twice, once for positive sentiment training data and once for negative sentiment data, and by having it as its own function it reduces code duplication.

4.3.3 Get Headlines For Model

This is the shortest function in the model, and much like the remove_noise function it is in a function because it is called twice, once for positive headlines and once for negative headlines. It is used only to generate the tokenised version of each tweet, which consists of the original tweet

altered such that it has had the stop words removed and all the remaining tokens normalised to lower case, and the sentiment of the original piece of data.

4.3.4 Headline Analysis

The headline_analysis function is used to ascertain the sentiments of each of the tweets that have been previously scraped and collate them for graphing in the later stages. It was extremely important to encapsulate this as a single module that could be called multiple times, since when I began working on the software I didn't know how many different stocks I would be using, and therefore how many times this part of the code would be needed, or indeed which timescale I wanted to use, and therefore how the variables should be named.

4.3.5 Data Map

The data_map function is similar to the headline_analysis function, in the sense that they are both called once for every stock being analysed and they are called one after the other. While this could have been achieved by combining the two into a single function, which would reduce the number of lines needed when calling, this was not done for two main reasons. Firstly, having these two different but connected actions contained in two separate functions made it easier to debug throughout the coding process, as it was much less of a challenge to isolate the problem to a smaller area. Secondly, any changes that were made to the code to make it more clearer or efficient were far easier to implement, since they were within a single small function that wouldn't cause errors to appear elsewhere in the program.

4.3.6 Auto Graph

The auto_graph function is the last function that is called when the program is run. It is used to create the graphs for each of the stocks listed, with each graph having the same design, labelling and size, to ensure that comparing graphs of different companies is as easy as possible. The function is called once for each of the stocks and takes 4 parameters, the lists that contain the positive percentage and net sentiments for that stock for each week, the name of the company and its stock code. The auto_graph function generates both variants of the graph for each company one after the other, which are the saved as PNG files to be viewed again later, without having to rerun the entire program. It was extremely important that this part of the code was contained within a single function that could be called with different parameters each

time, since the amount of duplicated code would have been massive, since beside the data that is actually being shown in the graphs nothing is being changed.

4.4 Third-Party Content

The project contains some third-party content in order for the requirements to be completed, as well as to dramatically decrease the amount of time that would be required to finish the project and to later on refine it.

4.4.1 Natural Language ToolKit Library

The project uses the Natural Language ToolKit library [2]. It is a Python library that is commonly used for natural language processing, and is used within the project as the basis for creating the sentiment analysis model and much of it is crucial for analysing scraped tweets.

4.4.2 SNScrape Library

The SNScrape library [7] is used to obtain thousands of tweets from different financial news Twitter accounts. It is an open source scraper that is used to create the csv files containing all the tweets mentioning and referring to the company specified. SNScrape was used as it allowed access to far more data than Twitter's own API, which since the development of the project has been restricted even further, making the decision to use third-party software to gather the tweets absolutely correct in hindsight. The only requirement needed to use SNScrape is having Python version 3.8 or later.

4.4.3 Twitter

Twitter is a social media platform that allows users to create accounts and post tweets. These accounts can be private, in which case only certain accepted users are able to see them, or public. For the purpose of the project all the accounts used were large media and financial news accounts, meaning that all the tweets are publicly available. All of the social media posts that were analysed for the purpose of this project were taken from Twitter, since although the same project could be completed using another social networking site, Twitter, as previously mentioned in the Background section of the report, is tailored towards the dissemination of news.

4.4.4 Yahoo Finance

Yahoo Finance is a financial news website that provides news, data and commentary on finance including stock quotes and financial reports. It also has historical data for all major companies dating back to their initial public offerings, showing the complete history of their share price in daily, weekly or monthly intervals. This website was the source of all of the share price data used by the project, since it is all free to use, easily accessible and can have the period and intervals easily defined by the user before downloading the data.

Chapter 5

Implementation

This chapter details the development of the software. The development approach chosen was an agile iterative approach. This was chosen because it allowed for the development of multiple versions of the software, each of which was a working implementation and that could then be improved, in terms of accuracy, reliability and scale and the project proceeded.

5.1 Version 1 - Initial Sentiment Analysis Model

As the focus of the project was on the use of Natural Language Processing to reliably analyse sentiment and then find patterns when compared to financial data, it made sense for the first part of the project to be completed to be the Sentiment Analysis model. At this stage, I chose to implement an existing model, specifically the Naive Bayes classifier from the Natural Language ToolKit suite. There were a couple key reasons for this. Firstly, there was plenty of literature about this classifier already available online, which would make the process of understanding how it works, how to implement it and how to use it most effectively within the project much easier. Additionally, Naive Bayes provides great accuracy especially with respect to how difficult it would be to implement. Lastly, the large amount of training data that was accessible, as there were many sentiment analysis data sets related to social media posts also made this the right decision, since one of the biggest advantages of Naive Bayes is how well it performs when given large amounts of training data [8].

I also downloaded several datasets to be used to train the Naive Bayes classifier. While there were many larger datasets used for sentiment analysis of social media posts, I felt that the best one at this early stage was a smaller dataset called ”Sentiment Analysis for Financial

News"[9], as although many of the other datasets were much larger and approximately 10% of the entries were unusable due to the fact that they had been assigned neutral sentiments, the fact that the dataset was so tailor-made for the type of data I wanted to be analysing meant I felt it was definitely the right decision. At this point the accuracy of the model was averaging between 75 and 80%, which I felt was more than acceptable for the first version of the model.

I also wrote an initial version of the code needed to scrape the tweets from a given list of Twitter accounts. I initially planned to use the official Twitter API for this purpose, however after further investigation I found that this would not be suitable, as the limitations imposed by Twitter, such as only being able to scrape tweets over the previous two weeks and limiting the number of tweets that could be scraped without paying a fee, made this untenable. In hindsight, this was an even better decision than I could have imagined, since as of Thursday 9th February, free access to Twitter's API was removed. I decided to use an open source scraping tool to access the tweets from these companies and settled on SNScrape. SNScrape is one of the most popular tools for scraping social networking sites, and is well supported and documented, which were the two main factors that led me to use it for this project. The initial version of the scraping software was hard coded to scrape tweets from three different accounts concerning only one company, as my only goal at that point was to make sure that the theory and concept of the project was plausible and executable.

I the used Yahoo Finance in order to download historical stock prices. I initially wanted to show the daily share prices on the graph, but after testing this and seeing that the graphs produced were extremely unclear and not at all useful for analysis, I switched to plotting the weekly stock prices, as this was still fairly detailed but also readable when converted into a graph.

From there I imported the Matplotlib library to generate graphs that could then be analysed to show patterns and correlations in the data between the percentage of positive tweets in the news related to the company and their own stock prices. As with the use of SNScrape tool, I wrote minimal code here to test whether the graphing would work so that I could expand and improve upon it as the project continued.

5.2 Version 2 - Improved Prediction Accuracy and Expanded Range of Stocks

At this point I began investigating other potential sentiment analysis models, but felt that the model itself would be perfectly acceptable for the project, so I elected to keep the model and instead improve it's accuracy as much as possible without recreating the entire thing. I was able to find a second test dataset, which allowed me to create a combined test dataset with approximately 4600 tweets to train and test the model with. I also changed the ratio of training to test data, increasing the number of statements used to train the model, which led to the model accuracy increasing to approximately 87% on average.

I also expanded the number of stocks being compared from 1 to 4, to begin to look at potential patterns and correlations between companies as well as between stocks and sentiments. This involved scraping 3 more companies tweets and downloading their historical stock prices from Yahoo Finance. It was at this point that I began to notice some severe problems when generating graphs with bizarre line shapes that were clearly incorrect.

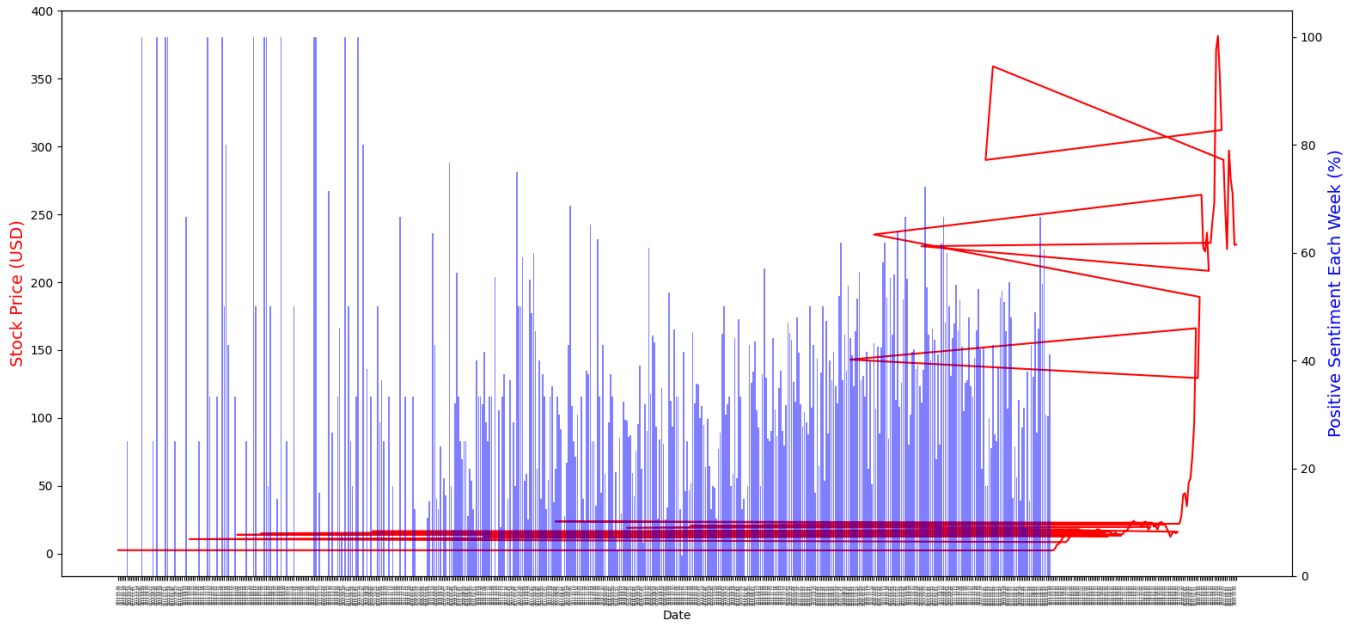


Figure 5.1: Example of graph where dates for stocks and sentiments did not correspond

After a lot of debugging, I realised that the issues were caused by the fact that the dates the stock prices correlated too were inconsistent with the dates separating different sets of

analysed tweets. This led me to make all the stock price dates consistent, with each stock and Twitter sentiment first being tracked on January 1st 2013 and every week since until January 2023. This was beneficial for a few reasons. Firstly, it gave a convenient time span to track all these companies of 10 years, which was long enough to visualise any medium level and macroeconomic trends. Secondly, it limited redundant scraping, since most major financial news media organisations were active on Twitter by 2013. Thirdly, by tracking all these companies identically, companies could be fairly compared with each other.

5.3 Version 3 - Increased Number of News Sources and Graph Clarification

After having improved the model and expanded it's training data to the point that I was happy with it, and settled on a consistent system when it came to categorising tweets and dating stock prices, I felt that I was ready to expand the project to more stocks and improve the graphing of the results.

I selected an additional 6 stocks to bring the total up to 10. This was incredibly easy to do, since the modular nature of the code meant that I only needed to download the weekly stock prices, create some variables to store the relevant stock specific values, and scrape Twitter for tweets pertaining to these new stocks.

As opposed to just scraping tweets for these new stocks, I decided to massively increase the number of financial news accounts that would be scraped, in order to improve the reliability of the end results by analysing the sentiments of news from many different sources, in an attempt to reduce bias as much as possible. This meant scraping tweets for the previous 4 companies again and brought the total number of tweets to be analysed to just over 110,000.

After running the model and observing the graphs that were generated, I decided to make some changes in an attempt to make the results much more easily readable and understood, as having the percentage of tweets that week that were analysed as being positive didn't seem to clearly show the relation between sentiments and stock prices. I created a second version of the graph, showing the net sentiment of each week as a bar against the stock price.

5.4 Testing

Testing is arguably the most important part of the development process of any software, as it ensures that the final product is functional and that all requirements have been met. For this project, the main concern was the accuracy of the results, since the code is such that each time it is run, the results ideally will be almost the same, and the graphs generated can then be used for future analysis. Other common important criteria such as security, speed or accessibility were not particularly important, since the project does not revolve around an application that may have many different users, rather a single script that can be run once to analyse large amounts of data, whose results can be used over and over.

5.4.1 Non-Functional Testing

Non-Functional Testing is a type of testing that checks aspects of software not explicitly defined in the requirements of the software. Here are the key aspects of Non-Functional testing and the extent to which the project satisfies them:

- Security: As previously mentioned, this project acts largely as a script that can be run to analyse data and then generate graphs for further analysis and evaluation and as such the security of the application is not massively relevant. The use of external libraries and requests, particularly when scraping Twitter could be argued to be a risk, however the csv files being generated have little to no capability to affect the security of the application or the system that is using it.

- Reliability: The software itself is such that, assuming no errors have been made in terms of naming and saving files such that they are accessible to the system, the application will always generate graphs the graphs it is supposed to. The minimal user inputs and little reliance on outside requests to the internet after the scraping has been completed once mean that, bar considerable user error, the software will always run correctly.

- Efficiency: The speed of the program is by far its biggest weakness. The amount of time it takes for the system to create, test and train the Naive Bayes Classifier, then analyse the well over 100,000 scraped tweets it is given and plot graphs is considerable. This normally takes 3-5 minutes, which is a considerable wait. This does not however include the amount of time required for the twitter_scrape function to run. Searching through many millions of tweets obviously is extremely time consuming, and this function normally takes at least 25 minutes, however this can be massively increased depending on the number of

accounts being scraped and the number of companies to be analysed. Once the accounts to be scraped and the companies to be analysed have been selected, this function only needs to be run once on the system as the CSV files that are generated can be reused every time the program is run. This means running the program for the first time normally takes around 30 minutes, which is not at all efficient, but the trade off and the fact that afterwards it will be significantly faster mean that this inefficiency is significant but not catastrophic.

- Flexibility: The program is simple enough that assuming the bare minimum system requirements are met, which in this case is just having Python 3 installed, the software will work correctly out of the box on any system.

- Modifiability: The software was designed with modifiability in mind, since the main goal is that this can be used for any company or stock to see the relationship between them and social media sentiment. As such, adding companies and stocks is relatively easy, as it only requires adding the name of the company in the twitter_scrape function, creating the two dictionaries as already shown in the code and duplicating the existing calls to the headline_analysis, data_map and auto_map functions replacing the parameters with those relevant to the new company.

- Testability: One of the other significant weaknesses of the project is its testability, which is linked somewhat to its efficiency. As it acts largely as a script, much of the usual testing methods aren't as applicable compared to an application designed for new users to interact with frequently. However the long waiting periods after running the code for results to be generated means that debugging and testing different inputs and parameters, for instance when trying to optimise the classifier, have very slow response times, making testing extremely time-consuming.

5.4.2 Functional Testing

In this subsection of testing, the final version of the project is compared to the list of requirements set out in chapter 3 of the report, with explanations of how each requirement has been met:

Twitter Data

- Twitter data should come only from verified financial news accounts: Verified financial news Twitter accounts were deliberately chosen and added to a list whose items were provided as parameters for the SNScrape implementation within the twitter_scrape function, and used to identify the account that tweets are to be scraped from.
- Twitter data consist only of tweets that contain the name of the company being analysed: Much like the previous requirement, the names of each of the companies that are to be analysed are added to a list, whose elements were provided as parameters for the SNScrape implementation within the twitter_scrape function, and used to only obtain the data of tweets that actually contain the name of the company in the body of the text.
- Twitter data should consist of the text of the Tweet and the date it was posted: The username of the account that made the tweet, the date that the tweet was posted, and its contents are saved to a dataframe.
- Twitter data should deliberately exclude replies and retweets: The twitter_scrape function contains an if statement that searches the content of each tweet that is found, and discards it if it is a retweet or a reply.

Financial Data

- The financial data should consist of the relevant share price information, such as opening and closing prices, high and low points, and the date from which the data was taken in a day / month / year format: The financial data is taken from Yahoo Finance and contains the share price for each week and the date for that week.
- The financial data should contain the relevant prices for the stock at weekly intervals from the maximum time frame or since 2013, whichever is shorter: This requirement changed slightly during development, as it became apparent that a 10 year period from January 2013 to January 2023 was much more useful for comparisons between companies.

Code

- The code should iterate through the test dataset and split each tweet into a list of individual words (tokenization), and separate the test tweets according to whether they have positive or negative sentiments: The first lines of the code that aren't contained within a function are used to split the test data into individual tokens.

- The code should remove special characters, punctuation, and links from the list of tokens: The remove_noise function removes all special characters and other unwanted tokens from a set of tokens.
- The code should convert each token to its normal form: After removing special characters, the remove_noise function uses the lemmatizer imported from the Natural Language ToolKit to convert each token into its normal form.
- The code should create, train, and test a sentiment analysis model using this tagged dataset: After the data is divided by sentiment and reincorporated and randomised, the Naive Bayes Classifier is created.
- The code should iterate through each week from 1st January 2013 onwards, and analyse the sentiment of all scraped tweets from that week, incrementing counters that track the total number of tweets, total number of positive tweets analysed and total number of negative tweets analysed for that week: The headline_analysis function is used to iterate through all of the tweets for a given company, look at the date that each scraped tweet was posted, determine the sentiment and then increment the number of tweets of that sentiment for that week by 1.
- The code should create a dictionary, with each key a week the tweets came from, and each value being the net sentiment of the analysed tweets for that week: The data_map function takes the results of running the headline_analysis function as parameters to create a dictionaries of weeks for each company as keys, and the percentage of positive sentiment or net sentiment number as values.
- The code should generate graphical representations of the data that has been collected: The auto_graph function contains all of the code related to constructing graphs, as it takes several key parameters whenever it is called.

Graphical Representations

- The graphical representations should contain two separate graphs. One should be a line graph showing the stock price each week, and the other should be a bar chart with each bar showing the net sentiment of all the finance tweets that were analysed that were posted that week: The MatPlotLib library is used in the auto_graph function to create plots that each contain two separate graphs that are laid on top of each other, which works because all the data has the same date values. The line chart containing the stock prices

is overlaid on top of the bar chart. Two different plots are created, one with contains the percentage of tweets that week that are positive, and one that shows the net sentiment towards the company that week. The graph containing the net sentiment is the one that is used for analysis due to the fact that low or negative sentiment is more easily understood in that graph, although any graph using that same data in a different way could be easily created within the function.

Chapter 6

Legal, Social, Ethical and Professional Issues

This project is based around two key concepts that are both extremely relevant to legal, social and ethical issues: Social Media and Stock Prices. The British Computer Society Code Of Conduct [10] was not massively relevant to the project due to it's limited scale and the fact that it largely acts as a script that generates graphs that can be analysed further by the user, however point A of the Public Interest section of the Code Of Conduct was important, as it talks about the need to have regard for privacy of others. Care was taken to ensure that social media data was only taken from large public news and media companies, and as such the privacy of individual was not breached. Similarly, the use of publicly accessible historical data regarding stock prices over the last 10 years limited the potential ethical issues with this project.

Additionally, it was important to make sure that the project was created ethically and as such, the project consists entirely of my own work, open-source libraries and some open-source code provided online, that has been altered slightly for clarity within the greater context of the rest of the code. While I initially wanted to complete the project without using any third-party content, as the project progressed this became an impossibility, as the amount of time it would take to construct both a sentiment analysis model and some code to generate graphs completely from scratch would mean that the project could never be completed within the time constraints. The usage of these open-source libraries and code is clearly stated and shown within the final code.

Chapter 7

Analysis

This section of the report will discuss and summarise the results and findings of running the finished software and generating the final graphs. This will involve deeper analysis and discussion of some of the graphs generated by the code, chosen as they can show the common patterns and correlations more easily. All of the graphs that were generated by running the final version of the software are available to look at in Appendix A at the end of the report.

The biggest takeaway found by examining these graphs is that if there is a period of negative sentiment of a month or longer, or at least a period of sentiment that is much less positive than is normal, is generally succeeded by a sudden and significant drop in stock price within at least the next month. There are often times when sudden increases or declines in the stock price overlap almost exactly with the same behavior in the sentiment, although I think this can be clearly and largely attributed to the sentiment of news articles reporting on a sudden increase or decrease, rather than the overall sentiment causing the change. These all seem to be becoming more obvious as the sentiments analysed become more recent, presumably as social media use increases then the accuracy of the sentiment effect also increases. It is also important to note that these periods of low relative sentiment often occur during periods of increase for the stock price, showing that there isn't a direct causal relationship between the sentiment and stock price that week, and that regardless of how well the stock is doing while the sentiment is below average, consistent negativity inevitably causes a drop at some point in the near future.

The validity of this does depend on the individual company that is being analysed, as although there is a fairly consistent pattern, in many cases it is much more consistent and pronounced, and therefore much more obvious and reliable. For example, the net sentiment graph generated for Starbucks had net sentiments that were far more consistently negative than

other companies that were analysed, but the continual periods of negativity consistently led to sharp drops in stock price.

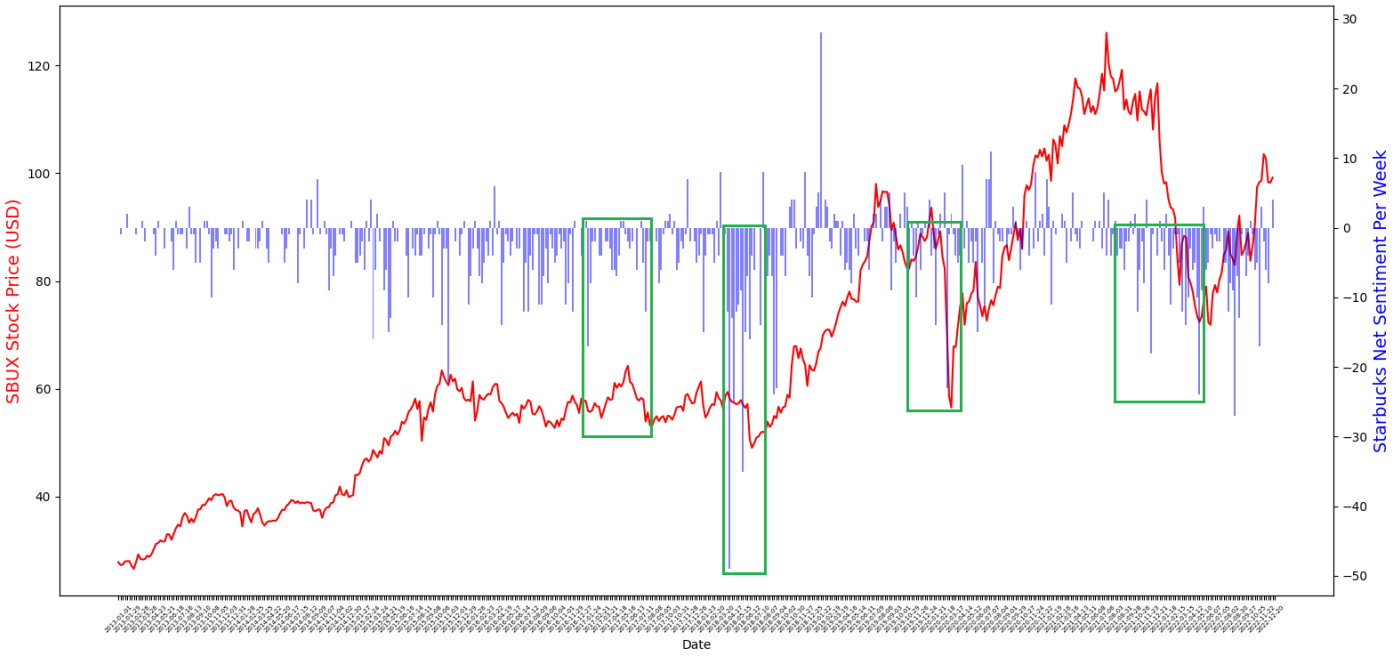


Figure 7.1: SBUX price against sentiment for that week, with periods leading up to sharp drops highlighted

The graph has few weeks where the net sentiment is positive, and even less of these positive weeks actually correspond to increases in the companies share price. The periods of notably low sentiment that occur shortly before a sharp decline in the value of the company are highlighted by the green box, which shows how consistent this pattern is in this case. It is worth noting here that the net sentiments for Starbucks have a much shorter range than many of the other companies analysed, with the highest positive net sentiment being less than 30 and the highest negative net sentiment just under 50. This shows that even when the graph shows a massive high or low, the positivity or negativity isn't actually as overwhelming as it may appear. This therefore reinforces the strength of the pattern relating low periods of relative sentiment to drops in the value of the company, since the correlation still seems to hold when the net sentiment doesn't actually have overwhelming numbers behind it.

This pattern can also be observed with a company that has a much more generally positive sentiment surrounding it, for example Disney. The social media sentiment surrounding Disney

was rarely a net negative, at it's lowest points the sentiment was almost completely neutral, with it being either slightly positive or slightly negative. In this case, periods of sentiment that are low in comparison to the rest of the graph are what signify an incoming drop in the share price.

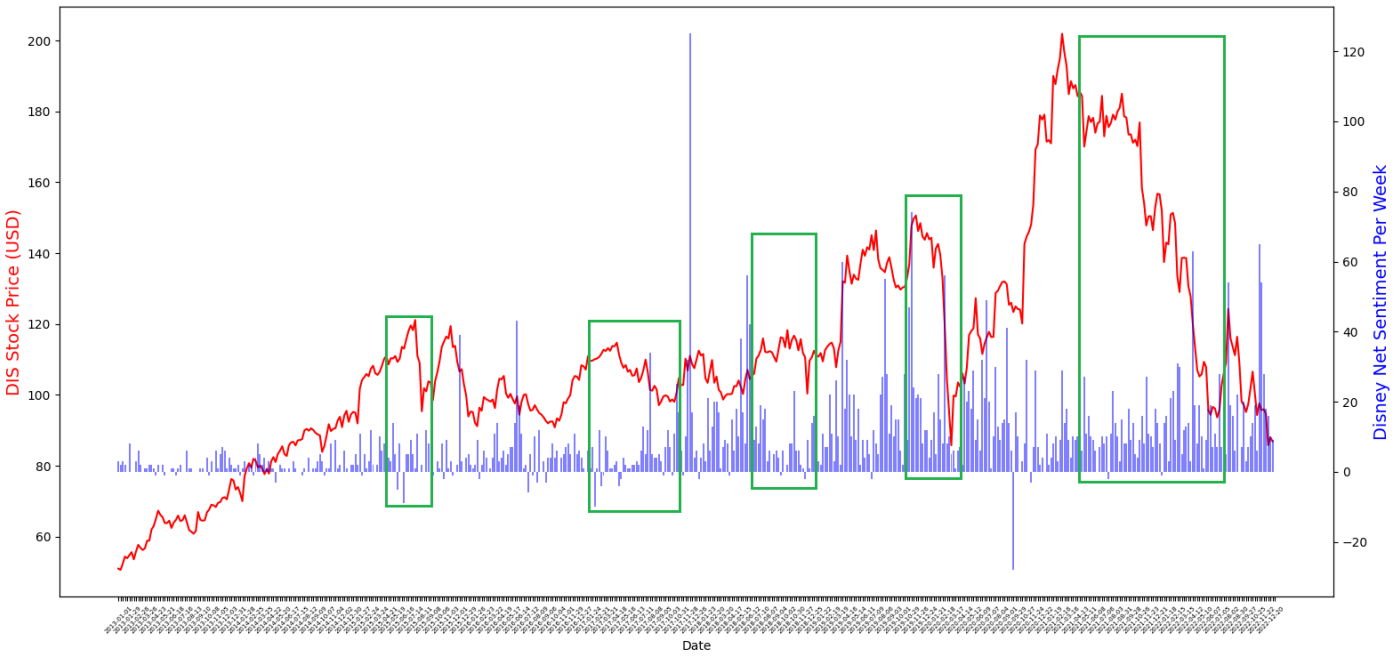


Figure 7.2: DIS price against sentiment for that week, with periods leading up to sharp drops highlighted

This example is slightly less clear due to the fact the overall sentiment was rarely negative, however the pattern of blocks of near neutral sentiment with occasional negative dips foreshadowing declines and sudden drops in the value of the stocks remains.

A third type of example is that of a company whose stock value growth is consistent, and has sharp drops that are both exceedingly small compared to other company graphs, and are also much more rare. In this case, the graph for McDonald's shows the fact that as a business they are as well protected or better compared to other businesses against economic downturns and changes, but the having a streak of low sentiment weeks still signals an incoming stock price decline.

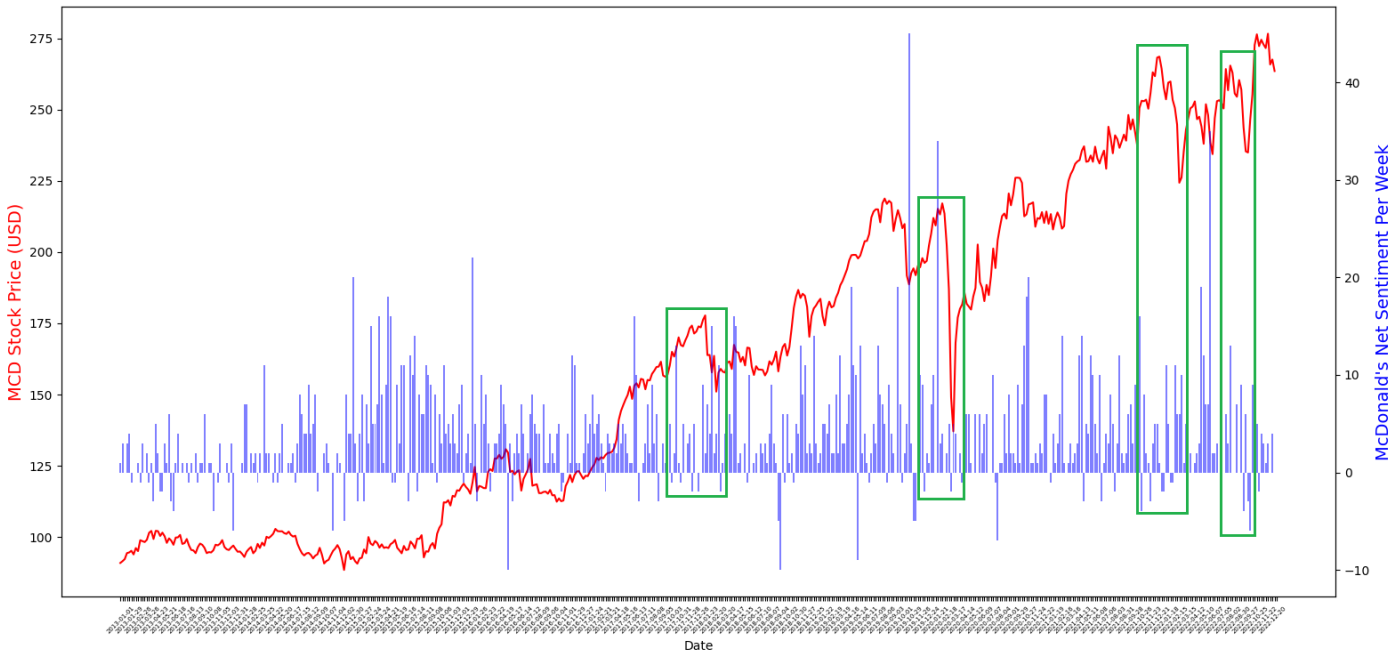


Figure 7.3: MCD price against sentiment for that week, with periods leading up to sharp drops highlighted

Here we can see that while the McDonald's share price chart shows few periods of either stagnation or decline, the sudden drops in share price that do occur are once again preceded by at least a few weeks or at most a couple of months of near neutral and negative sentiment towards the company.

A slightly different variation of this pattern is evident in the analysis of the graph generated for Amazon, where the stock price chart consists of consistent growth and the occasional tiny drop in value, before a massive and sudden decline in value recently. This would seem to suggest that companies that almost never have sharp drops in stock price still have periods of low relative net sentiment before their sharp declines.

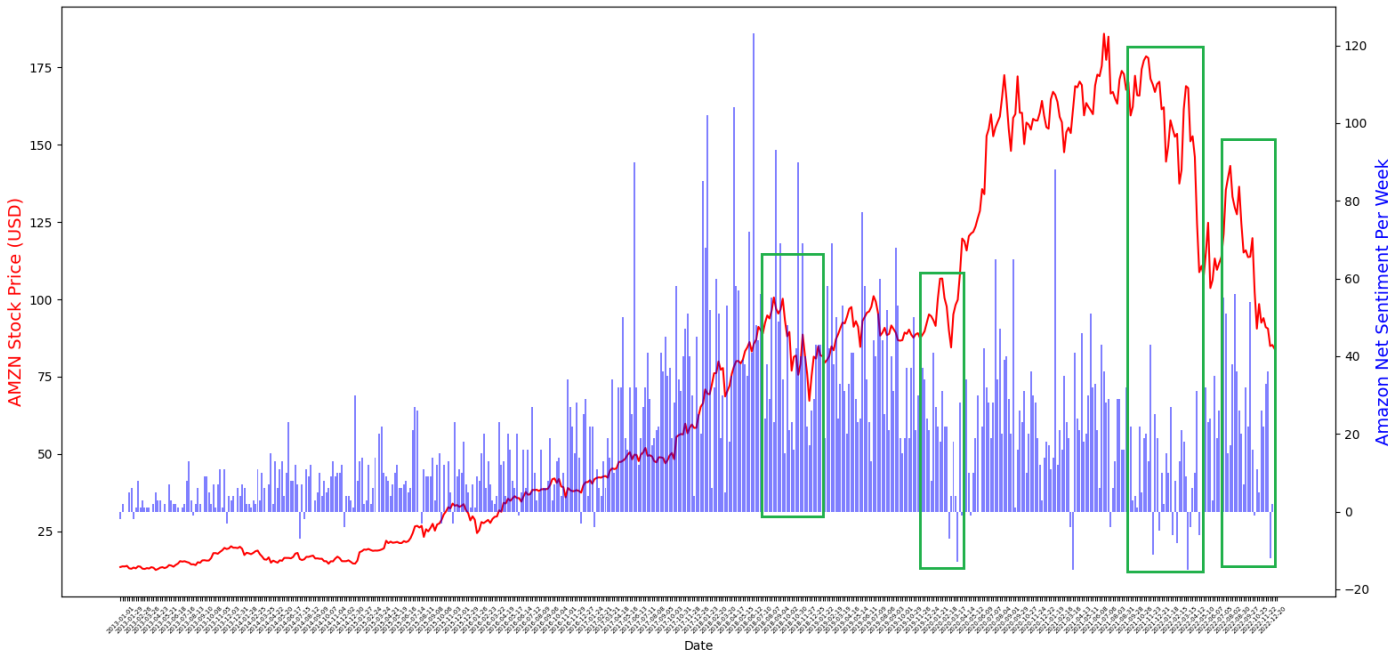


Figure 7.4: AMZN price against sentiment for that week, with periods leading up to sharp drops highlighted. Note the large drop at the very end of the chart.

It also highlights clearly the fact that periods of low relative net sentiment don't guarantee that a sharp drop in value is imminent, but that the sharp drops, whether they are fairly common or extremely rare, still follow prolonged periods of relative negativity on social media. The data in the graph from 2013 to 2018 shows this, as it begins with long periods of sentiment that would be considered low when compared to the rest of the graph later on, as well as short outright negative sections. These are present despite the consisted rise in stock price, which is far from flat but contains no major valleys showing a change in the outlook from investors at large towards the company.

This is also applicable for stocks that spent much of the time period in relative obscurity to the general public, then suddenly became well known to casual investors. Pfizer is the perfect example of this, as despite being a billion dollar multinational company that is nearly 200 years old, it was not being discussed online much before 2020, with well over 5500 of the 6500 tweets referencing Pfizer being posted after 2020. This is not surprising, as the beginning of the Coronavirus pandemic in 2019 had shifted the focus of the entire world towards the disease and attempts to develop a vaccine, and Pfizer's successful trials and eventual release of a vaccine in

2020 immediately made them a well known company worldwide, with this fame and importance related to stopping the pandemic giving them massive amounts of attention from news media accounts on Twitter.

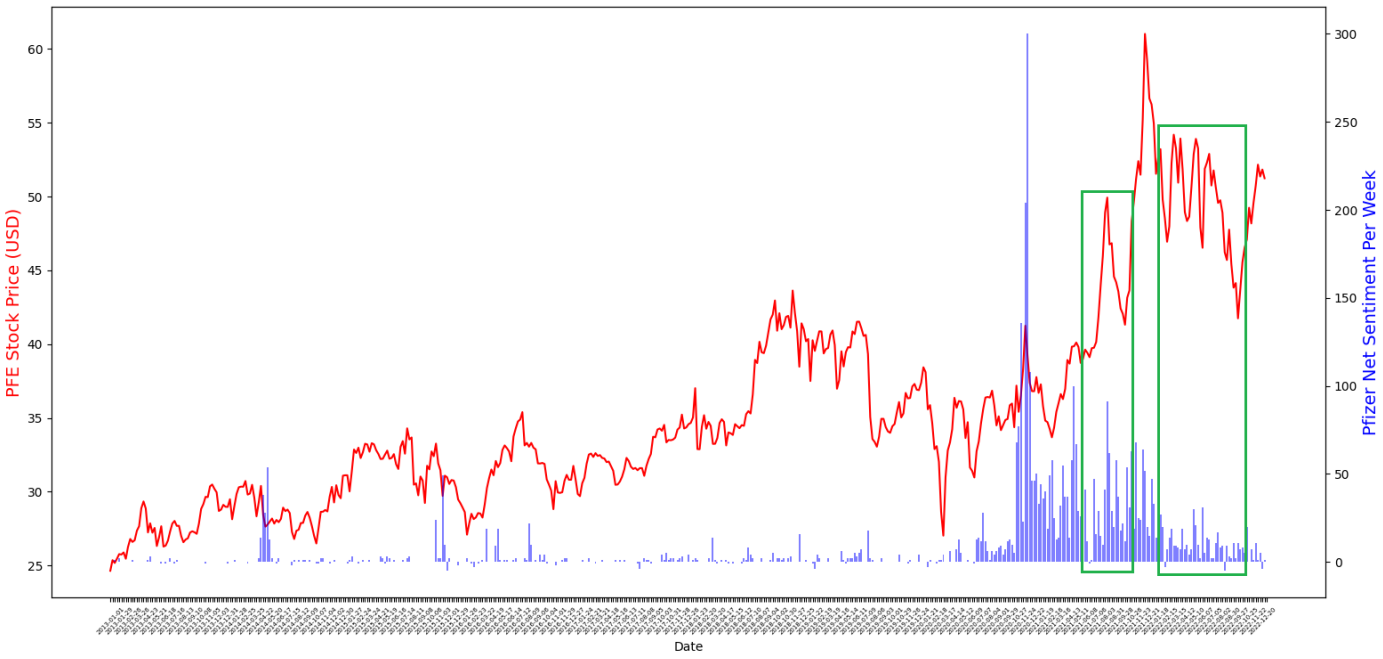


Figure 7.5: PFE price against sentiment for that week, with periods leading up to sharp drops highlighted. Note the large drop at the very end of the chart.

Here we can see that the first 7 years of the graph contain very low net sentiments that do not seem to have any correlation at all with the share price line chart, which contains its own fair share of sharp rises and falls in value. This seems to reinforce the idea that in order for social media sentiment to be a valid predictor of future changes in the share price of a company, there needs to be a considerable number of posts over a long period of time, as having less than 1000 social media posts mentioning Pfizer over a 7 year period provides far too little data to actually make a prediction, let alone for it to be accurate or useful. It is also clear from the graph the moment at which the company became much more well known, with a large spike in sentiment. From this point onwards, there is a much higher number of posts related to Pfizer, and two examples of periods of relatively low net sentiment preceding sharp drops in value, which can in this case be inferred to represent the overvaluation of Pfizer after developing a Coronavirus vaccine and them slowly being revered less by the media and public at large, since

the sense of reliance on this pharmaceutical company has declined as the pandemic has drawn to an end.

The graph generated after the analysis of tweets related to Apple is also consistent with this pattern, with there being four notable drops in share price preceded by low sentiment periods. Apple is another company that rarely had net negative sentiment towards it on Twitter, but had largely low positive weeks with the occasional massive spike in positivity.

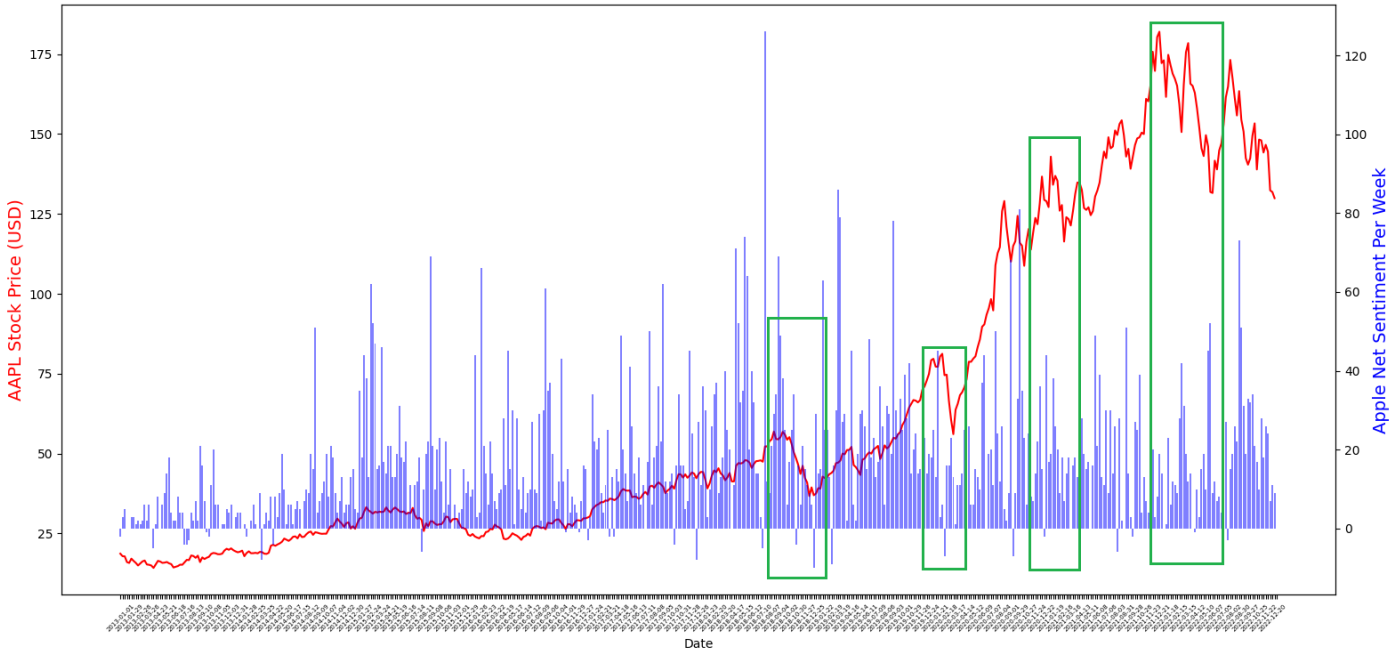


Figure 7.6: APPL price against sentiment for that week, with periods leading up to sharp drops highlighted.

These occasional negative net sentiment weeks often coincide with the relatively low sentiment periods preceding sudden drops in share price.

The results for Netflix are arguably the ones that least coincide with the observed hypothesis of low periods being predecessors to sharp declines in value, however the pattern is still observed with one key difference. This difference is that many of these low periods also contain the occasional spike in positive sentiment in the middle of these low periods.

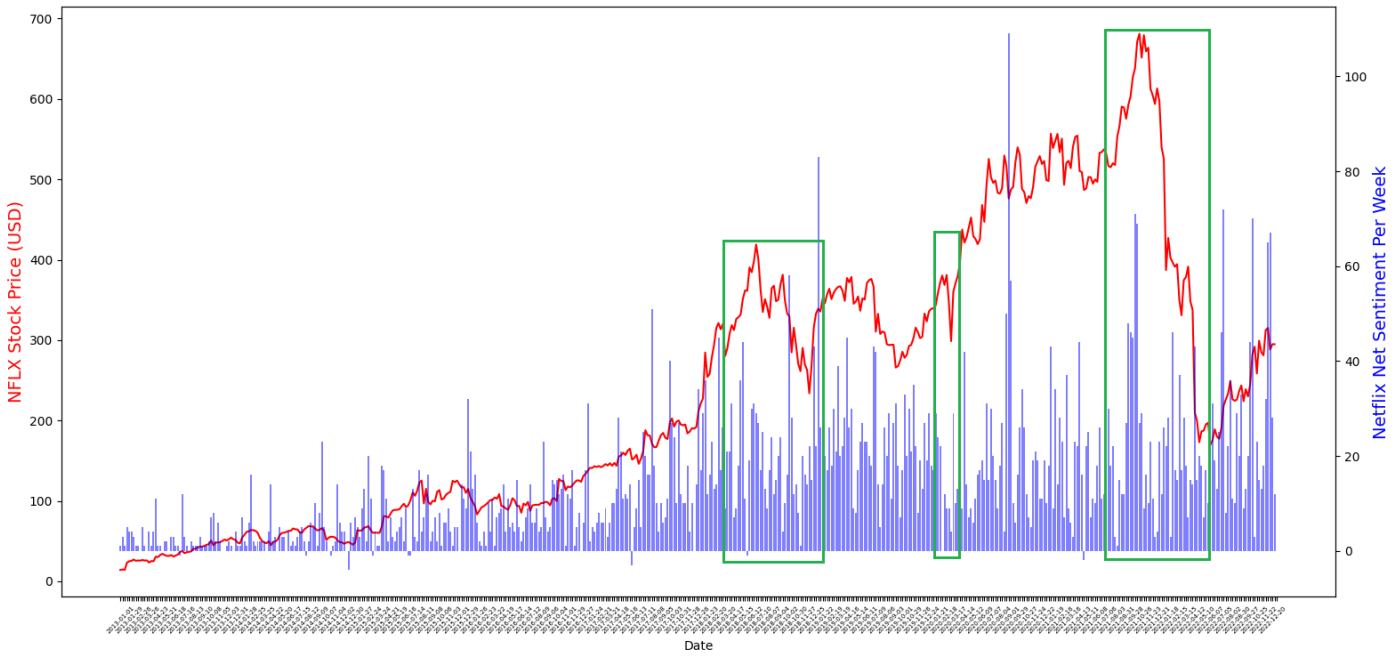


Figure 7.7: NFLX price against sentiment for that week, with periods leading up to sharp drops highlighted.

The final and sharpest decline in the graph above shows this very well. It begins with a few weeks of clearly low sentiment followed by several weeks of relatively high sentiment as the stock price rises again. This is followed by low sentiment right before the steep decline and continuing low sentiment throughout the two sharp drops that follow. Much like the graph for Apple shown previously, the stock price line chart has very few steep drops in price and has been on a relatively steady incline, barring the one major steep drop previously mentioned, since 2013.

The overall inference that can be made seems to be that the overall future of a particular stock, as in it skyrocketing in value and remaining there or dramatically falling and never recovering can never be reliably predicted by social media sentiment, at least not in the weekly intervals that I have been tracking it with, since there are far too many external variables that aren't able to be taken into account or even acknowledged when the data that is being analysed consists of short micro blogs online. However, the analysis does seem to suggest that there is pattern that could be used to make profitable short term trades, that prolonged periods of negative or rather, more negative than usual, sentiment can reliably predict a future drop in

value, within approximately three months of the period beginning. With further analysis and testing, and extrapolating forward into the future, this correlation could be used to identify the right period to sell a stock, right before a sharp drop in value, and buy it again once it has seemed to bottom out and begin to steadily rise once more.

Chapter 8

Evaluation

This section of the report will evaluate the ways in which the project could be improved.

8.1 Sentiment Analysis

Choosing to import an existing sentiment analysis algorithm from the Natural Language ToolKit was the first major decision made in this project. Although it would have been possible to create a sentiment analysis algorithm from scratch in order to process the many tweets that were scraped, the fact that I had no idea of how to actually start creating an algorithm like that or how much time it would take meant that the risk of not completing the project was too high. By implementing an existing library and using online examples to learn to perform sentiment analysis, I was able to reduce the risk involved and spend more time focusing on gathering data, improving accuracy and generating useful graphs. If time was not a factor, I would have liked to create a sentiment analysis model from scratch using a more advanced and more accurate algorithm; however, the accuracy of the model for this project was still more than acceptable to be worth analysing.

8.2 Graph Generation

One of the key decisions made early on in the development of the project was deciding the intervals at which sentiments and price of shares would be recorded. I originally settled on weekly intervals because it was the shortest interval that, after adjusting many of the settings in my MatPlotLib implementation, was readable and understandable for the end user. However, when it came to analysing the results later, making some adjustments to the code and

downloading historical share prices at daily rather than weekly intervals would have allowed for a more precise analysis of the relationship between share prices and social media sentiment. This would have required significant work to actually be implemented and even more to attempt to configure the graphs so that they would still be understandable, as the difference between 520 and just over 3650 data points is considerable with regard to readability.

8.3 Application Speed

The most immediate improvement I would want to make to the project would be to try and improve its speed and efficiency. The long wait times are the largest weakness of the project and the only real issue that makes it less effective as a tool for analysis. This is due to the significant amounts of data that need to be analysed, not including the massive time loss incurred when scraping tweets for companies. This time can obviously be reduced by decreasing the number of companies being analysed as well as decreasing the number of financial news accounts that are being scraped, however this comes at a cost of having less companies analysed and a significant reduction in the accuracy of the results. The final version of the software takes roughly 30 minutes to scrape Twitter. This is because it is scraping tweets from 16 different verified news accounts related to 10 different companies. Reducing the number of accounts or companies would obviously speed up the process, however it would therefore reduce either the depth or breadth of the scraped data. In order to speed up the running of the application without losing accuracy of the results, multiprocessing or multithreading could be investigated as a way to speed up both the scraping of data and the analysis of the tweets in order for the utility of the application to be considerably improved, as right now waiting around for 30 minutes at a time makes the software far less useful compared to how it could be.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

This project is focused on attempting to quantify the relationship between sentiment and value as accurately as possible, in this case abstracting sentiment to mean the attitude of news related to a company and how that affects its perceived worth in the eyes of investors and the public at large. Every objective and aim of the project at the beginning was completed, as the end result was a coherent analysis of the effect that social media posts, but specifically their sentiments, have on stock prices, as shown by the graphs that were generated and discussed in the Analysis section of the report. That said, the project is far from perfect and could be massively improved and expanded. The project relies heavily on the Natural Language ToolKit library and implements it to classify tweets as either positive or negative based on several different online examples. Choosing to build off of existing work was a key decision early on that allowed me to actually complete the project, but with more time and now with an understanding of how natural language processing works after spending so much time working on it, this could be replaced by my own version from scratch that would mean this project is much more independent and relies less on existing work. The classifier used in the project, Naive Bayes, while it is seen as one of the more simple classification algorithms, has shown itself to be efficient and accurate when categorising large amounts of data.

9.2 Future Work

There are several ways in which this project could be taken further, which due to the fact that the project was designed and written with a modular structure in mind, would not be too challenging to implement.

One of the first improvements that I would make would be to alter the time scale used when gathering the historical data about the different companies, separating tweets and graphing the results. While for the purposes of this project having weekly stock prices and the net sentiments of tweets separated by the week they were posted made sense and allowed for graphs to be created that were in enough detail that meaningful assumptions and later conclusions could be drawn from them, but not so detailed with such small time frames that in the end they were unreadable and therefore useless as a tool for analysis, a future improvement to the project would be having daily sentiments and stock prices graphed. By having smaller time frames for the x and y axis data, the graphs would be able to convey more detail, making them more accurate and also potentially showing trends over much shorter time periods that may be more useful and reliable. By digging deeper into the graph generation and spending some more time testing it would be possible for graphs with many more individual data points to be readable which, after making some slight alterations to the existing code and downloading historical data from Yahoo Finance at daily instead of weekly intervals, may uncover short term trends that would be useful when it comes to making intelligent investment decisions based on patterns of social media sentiment in the future.

The easiest way in which this project could be taken further would be to increase the amount of data being used, due to the fact that the important code used for analysis and graphing is at this point complete, the simplest way to improve the accuracy of the results would be to increase the amount of data it is given. There are two main types of data used within the project that could have their volumes increased to improve the accuracy and reliability of the results: the strings used to train the sentiment analysis model and the tweets that have been scraped from Twitter and will be used to show the relationship between sentiment online and the perceived value of the company being discussed. By searching for more training data to use, and making sure that the training data resembles the tweets that it will be used to analyse later, the sentiment analysis model will be able to more easily and accurately separate tweets that are positive towards the company being referenced and those that are negative.

Additionally, by increasing the number of tweets being scraped, in this case by adding more accounts, specifically accounts interested in news and more precisely financial news at least to

some extent, the reliability and usefulness of the results as a guiding tool for predicting and capitalising on financial trends will increase. It seems evident that having more tweets scraped will improve the accuracy of the results, because the higher the amount of data that has been scraped, the closer the net sentiment found will be to the actual attitude of people towards the stock at that time. A more understated aspect of this is the importance of gathering tweets from as many different sources as possible. Increasing the amount of information from differing sources will improve the overall value and reliability, since although the accounts are all reporting news, the subtle differences in bias would complete the picture in terms of showing all the attitudes towards the company and how they relate to its value.

At the same time, expanding the number of companies being analysed beyond the current 10 would be an easy improvement to the project, since all it would require would be slotting the new company into the existing structure of the code and downloading the historical data for that stock from Yahoo Finance. This would benefit the results of the project as it would either serve as more evidence that the pattern of extended periods of low sentiment with respect to the average for that specific stock is an indicator of an imminent decline in the value of that company, provide a counter-example suggesting that this pattern only occurs under more specific circumstances, act as evidence of a previously undiscovered trend that may be just as interesting or some combination of the above. Either way, any new insights gained will improve the usefulness of the project as a predictor of future stock market activity, meaning that expanding the stocks being analysed would be an addition that would require minimal additional work being completed and that could improve the effectiveness of the project massively, largely due to the fact that the structure of the project up to this point is such that making alterations and expanding the data is easy, because of the modular structure used during the design stage.

References

[1] H. Fathim. Sentiment analysis of news tweets. [Online]. Available: https://scholarworks.rit.edu/cgi/viewcontent.cgi?article=12198&context=theses

[2] S. Bird, E. Loper, and E. Klein. Natural language processing with python. o'reilly media inc.

[3] L. Nemes and A. Kiss. Social media sentiment analysis based on covid-19. [Online]. Available: https://www.tandfonline.com/doi/epdf/10.1080/24751839.2020.1790793?needAccess=true&role=button

[4] IG. Share price definition. [Online]. Available: https://www.ig.com/uk/glossary-trading-terms/share-price-definition

[5] P. R. Center. One-quarter of users produce the vast majority of tweets from u.s. adults. [Online]. Available: https://www.pewresearch.org/internet/2021/11/15/the-behaviors-and-attitudes-of-u-s-adults-on-twitter/pdl_11-15-21_twitter-0_9/

[6] O. U. Press. Natural language processing. [Online]. Available: https://oed.com/view/Entry/255272

[7] JustAnotherArchivist. Snscrape. [Online]. Available: https://github.com/JustAnotherArchivist/snscrape

[8] K. C. Lee. Sentiment analysis — comparing 3 common approaches: Naive bayes, lstm, and vader. [Online]. Available: https://towardsdatascience.com/sentiment-analysis-comparing-3-common-approaches-naive-bayes-lstm-and-vader-ab561f834f89

[9] A. Sinha. Sentiment analysis for financial news. [Online]. Available: https://www.kaggle.com/datasets/ankurzing/sentiment-analysis-for-financial-news

[10] B. C. Society. Code of conduct for bcs members. [Online]. Available: https://www.bcs.org/media/2211/bcs-code-of-conduct.pdf