Cardiff University
School of Computer Science and Informatics

Final Report
Final Year Project
CM3202 – 40 Credits

Sentiment Analysis of Financial News Headlines with Market Comparison

Author: Mr Harvey Allen *Supervisor: Professor Irena Spasic*

11/05/2022

1 Abstract

Growth of the internet and the digital economy, along with technical advances in computer and data science have supported a wave of alternative data sources that can be used to measure and predict the financial markets. One of these non-traditional metrics is public opinion mining, commonly referred to as sentiment analysis. This study investigates the hypothesis that the sentiment of financial news headlines reflects and directs the performance of the U.S. stock market through proving a significant correlation between the polarity of the sentiment and the change in price of a security, thus working to disprove the controversial 'efficient market hypothesis'. To evaluate the publics sentiment a vast dataset of 'financial news' headlines are required ranging over a broad period. Additionally, a natural language processing and machine learning classification model is built to predict the sentiment polarity of headlines. Finally, statistical analysis is conducted on the data to prove any significant correlation within the results. The study can demonstrate the hypothesis to an extent, showing that the sentiment of financial news headlines relating to the overall U.S. market, directly reflects the price of the U.S. market index. Despite this no correlation between the price of an individual stock and the sentiment of directly relating financial news headlines could be found. Additionally, there was no evidence to suggest that the daily sentiment of a security had any influence over its corresponding price change the subsequent day.

2 Acknowledgements

Thank you to my father and mother, James, and Dawn, for their support and guidance throughout my schooling and any adversity I have faced.

&

Thank you to my supervisor Irena Spasic for her continued enthusiasm in my work and driving my curiosity. Her assistance was indispensable in the success of this project.

3 Contents

4 Table of Figures

5 Introduction

Alternative data refers to non-traditional datasets that an investor can use to guide their investment strategy and portfolio. In addition to traditional metrics alternative data can provide insight into a securities earnings, economy, expectation, and emotions. This project will focus on market sentiment, the collective attitude towards a security or market, and investigate the hypothesis that '*the sentiment of financial news headlines reflects and directs the performance of the U.S. stock market'*. In turn this could help to prove or disprove the efficient market hypothesis and random walk theory and will function under the assumption that these widely criticised theorems do not hold true.

Consequently, the study will investigate securities within the S&P500 (A stock market index tracking the performance of 500 large companies listed on stock exchanges in the United States) as well as the broader index fund. Inherently the financial news articles will be targeted at the corporations encompassed by this fund, and it will provide the greatest reflection of the U.S. economy at a given time, under the assumption these publications are directly referencing U.S. securities.

Objectively this project can be divided into four core areas: data collection, data processing, data analysis and data visualisation. These four areas should all be met assuming they have been completed under the time constraints of this study.

Initially, the data will need to be collected. This project will be based on data scraped from CNBC, the Guardian, and Reuters official websites. Each dataset contains the headline, the last updated date and CNBC and Reuters contain preview text of the articles. Stock data will be collected from an API. The data will include close prices of the stocks during the analysed period. All the data collected will be cleaned for consistency and to remove any anomalous or missing data values. Exploratory data analysis will be performed for initial investigation, to spot anomalies and discover patterns that can affect later stages of the project.

Secondly, pre-processing will take place to make the data suitable for analysis, reduce processing time at later stages of the pipeline, and reduce dimensionality in feature extraction. This will involve out-of-vocabulary removal, stemming, lemmatization, stop word removal and TF-IDF. Additionally, the data will undergo information extraction and named entity recognition, which will be used to remove noise from the dataset. In turn this will reduce overfitting within the training data towards specific securities.

Sentiment Analysis will be used to determine the sentiment of each headline. Broadly, machine learning techniques will be used to determine the polarity of a headline. A training set corpus will be produced with rows representing independent feature vector containing information about a specific document (Headline), particular words and its sentiment, relying on public judgement for annotation. This training data will then be used to supervise the algorithm, to form the predictive model, based on a tested optimal classification technique to form the classification probabilities. Multiple supervised classification techniques will be tested against a series of evaluation measures to find the optimal model for this unique problem. Once this model is formed and can successfully calculate the sentiment of the

headlines the model will undergo hyper parameter tuning to optimise its metric scores further. Other techniques such as standardisation and cross validation will also be implemented to improve the fitting of the model. The pipeline will then be constructed, pre-processing, extracting features and classifying each headline passed through.

Finally, once the entire dataset has been processed through the pipeline, statistical analysis will be performed to determine the correlation coefficient between the sentiment and the price of the given security. The correlation will be calculated for both the markets reflection and direction. If deemed statistically significant, using a hypothesis test, the hypothesis can be rejected or accepted if there is a p-value that meets the acceptance level. Two visualisations will be produced to aid this analysis, a time series representing the reflection of the market price and sentiment monthly as well as a scatter plot and linear regression line demonstrating the direct influence the sentiment has on the market over a daily basis.

In summary, the results and visualisations will provide insight and understanding into the drivers behind the market's gains/ losses and in turn address the outlined hypothesis. This research will benefit professional investors, analysts, and portfolio managers in the utility of alternative data sources and what measures can be taken to gain valuable financial metrics from subjective natural language surrounding the markets.

6 Background

6.1 The Problem

The problem can be defined as proving or disproving the hypothesis that '*the sentiment of financial news headlines reflects and directs the performance of the U.S. stock market'*. The motivation behind this is to evaluate whether the sentiment portioning to financial news headlines can be used as signals for algorithmic or traditional trading. In the wider context of the project this will involve proving a direct correlation between the movement in a securities price and the sentiment polarity in a financial news headline. To achieve this over the scope of the project for an accurate result, a sentiment analysis classifier will be built through the utilization of a machine learning model. Constraints on the approach proposed suggest that an accurate machine learning model will need to be chosen and optimized due to the 'No Free Lunch Theory' (Wolpert 2002). This process will involve evaluating and testing several supervised machine learning classification algorithms against multiple metrics.

6.2 The Stock Market

The stock market is a venue where buyers and sellers meet to exchange equity shares of public corporations. A common method to raise capital for small companies is by selling shares of their business to investors in order promote expansion. Shares of the company are referred to as stocks. Investors buy stock with the intention of making money through either an increase in the price of their shares or dividend pay-outs. Supply and demand are the two core factors that influence the price of a stock, however these in turn depend on fundamental and technical factors. A core one of these factors being the market sentiment.

6.2.1 Efficient Market Hypothesis and Random Walk Theory

The Efficient Market Hypothesis (EMH) was proposed by economist Eugene Fama in his PhD dissertation in 1965. The EMH states that within an efficient market, asset prices reflect all available information. A direct implication being that it is impossible to 'beat the market' on a risk-adjusted basis since markets should only react to new information, thus excess profits cannot be made as securities are already accurately priced. Validity of the EMH has been questioned by many investors on theoretical and empirical grounds including Warrant Buffet, whose investment strategy and performance contradict the EMH and additionally argues that consistent performance of several funds cannot be entirely chance based (1984). Proponents, however, argue that outperforming the market is done not so out of skill but luck, as some portfolio managers will always outperform the mean due to the laws of probability.

In parallel, Random Walk Theory, popularized by Malkiel (1937) in his book 'A random walk down wall street', suggests that change in security prices have the same distribution and are independent of each other and that it is impossible to outperform the market without assuming additional risk. As news is unpredictable it is suggested that asset pricing follows a 'Random Walk' with a 50% chance of either going up or down, as ultimately an investor is either going to buy or sell. A notable

example of random walk theory occurred in 1988 that put to test Malkiel's claim that "a blindfolded monkey throwing darts at a newspaper's financial pages could select a portfolio that would do just as well as one selected carefully by experts", Wall Street Journal staff members played the role of 'dart-throwing monkeys' in the Wall Street Journal dartboard contest and won 55 of the contests compared to experts winning 87. However, experts were still only able to beat the Dow Jones Industrial Average (DJIA) in 76 contests.

6.3 Sentiment Analysis

Sentiment analysis, a sub field of natural language processing (NLP), is a growing field at the intersection of linguistics and computer science that attempts to automatically determine the sentiment contained in text, as defined by Taboda (2016). A sentiment is a subjective expression that describes an individual's feelings towards a particular subject or topic, differentiating itself from emotion which describes a phycological response. This study concerns itself with the sentiment polarity contained in financial news headlines towards a specific security or the broader market. Sentiment polarity can be determined as positive, negative, or neutral and describes the orientation of sentiment within text. Broadly, sentiment analysis can be utilized when dataset is too large to be processed by humans manually, such as political beliefs expressed over social media, customer product feedback and news articles.

Sentiment Classification:

- **Amplifiers** - An amplifier, also known as an intensifier, is an adverb that increases the polarity: "I *really* despise google".

- **De-amplifiers** - A De-amplifier, also known as a, down-toner, is an adverb that decreases the polarity: "I *barely* like amazon".

- **Negators** - A negator is a word expressing negation such as 'not', which reverses the polarity of a statement: "I do not like Meta".

- **Adverse Conjunction** - Adversative conjunction expresses opposition or contrast between two statements such as 'however', this increases the polarity of a word if it appears prior and decreases the polarity if it proceeds.

Currently, automated sentiment analysis is not as accurate as if it was performed by a human as each word is classified in a sentence separately by determining the polarity of each word is and how they affect each other. A negative word proceeded by a positive word could be classified as highly positive to a human but neutral to a computer due to their weighted average. Therefore, a core challenge of the area is analysing how words interact. Other challenges include polysemy, idioms, entity names and the evolving use of language.

6.4 Machine Learning

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving in accuracy given time. The key disparity between ML models and computer programs is the lack of intervention required by developers to instruct the system. The last decade has seen increasing popularity in ML, enabling both organisations and individuals to have insight and understanding into their dataset. ML algorithms take input data and use it to learn for themselves, without human intervention, through a sequence of statistical processing steps where they are trained to discover patterns in vast volumes of data. The data can be in the form of numbers, words, and images, given it is stored digitally. Sentiment analysis models can be trained to automatically process text data and understand it as a human would.

There are several techniques and complex algorithms used to command and train machines to perform sentiment analysis. These include but are not limited to:

- **Naïve Bayes** is a simplistic group of probabilistic algorithms that, for sentiment analysis classification, assigns a probability that a given word or phrase should be considered positive or negative (described further in section 7.37).

- **Linear Regression** is a statistical algorithm used to predict a Y value, given X features. Using ML, the datasets are examined to show a relationship. The relationships are then placed along the XY axis, with a straight line running through them to predict future relationships.

- **Logistic Regression** it is a predictive analyst algorithm and based on the concept of probability. A logistic regression model is like that of linear regression but uses a more complex cost function, this cost function can be defined as the ‘Sigmoid function’ (described further in section 7.37).

- **Support Vector Machines (SVM)** is like linear regression but more advanced as it implements a hyperplane and more complex data can be viewed multidimensionally (described further in section 7.37).

- **Random Forest** consists of several decision trees where training data points are split into random samples when building the trees. Trees learn from unique samples of data and are trained in parallel.

- **Deep Learning** aims to calculate data as the brain does using an artificial neural network. Deep learning Is a hierarchical ML model as it allows multiple algorithms to be used progressively.

The core challenge in ML is selecting the correct algorithm for the problem you are trying to model and optimising it for accuracy. This adversity can be further hindered by non-representative, poor quality or lack of training data. This can result in bias, inaccuracy, and poor fitting.

6.4.1 No Free Lunch Theory

The no free lunch theorem is a theoretical finding that suggests all optimization algorithms perform equally well when their performance is averaged over all possible objective functions. This implies for supervised ML that all algorithms are equally effective across all possible prediction problems (Wolpert 2002). This suggests that there is no predefined optimal algorithm to implement for this study and one will have to be found through analysis and testing. This is true as every ML algorithm makes priori assumptions about the relationship between features and target variables for a ML problem. Thus, the performance of a ML algorithm on any given problem depends on how well the algorithms assumptions align with the problem's reality.

The theorem states that given a noise-free dataset, '*for any two machine learning algorithms A and B, the average performance of A and B will be the same across all possible problem instances drawn from a uniform probability distribution'*.

6.5 Correlation and Causation

Correlation describes an association between variables. A correlation is a statistical indicator of the relationship between variables. These variables change together: they covary. But covariation isn't necessarily due to a direct or indirect causal link.

Correlation Coefficient Equation:

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{\left[n\sum x^2 - (\sum x)^2\right]\left[n\sum y^2 - (\sum y)^2\right]}}$$

Causation implies that changes in one variable bring about changes in another: a cause-and-effect relationship. These variables are correlated with each other so there is a casual link between them.

Correlation doesn't imply causation, but causation does imply correlation.

It is important to differentiate the two, as variables may change with each other more regularly than other however that does not mean a significant relationship. In this study we will analyse the correlation between the sentiment of financial news headlines surrounding a security and the price of that security and utilize a significance test to prove or disprove a hypothesis.

6.6 Stakeholders

Identification of patterns and insights shapes an investors' day-to-day and dives his or her investment strategy. The research plays a vital role in placing the insights generated from market data and breaking news into a wider context, it provides investors with the perspective to understand the importance of the current affairs information they are handling while also providing a framework for decision making. Any correlation proved within this work could fuel further research into alternative data investment strategies or cause implementation of a sentiment based algorithmic trading strategy. A functioning sentiment classification pipeline for financial news

headlines could also be used as a gauge for economists to evaluate the impact of specific events on individual's opinions as well as various financial metrics.

Quantitative hedge funds would be the core beneficiary of this work as signals can be extracted from the sentiment and incorporated into a financial model. However, as alternative data continues to rise in popularity within the financial world, this work could have influence within mainstream funds and the banking system.

6.7 Related Work

Recent studies have shown that alternative datasets appear to be entering the financial mainstream as the number and diversity of readily accessible alternative datasets has ballooned in the last decade and are no longer being used by just specialist players such as hedge funds. Data sources such as satellite imagery, social-media streams and consumer microdata are all providing an edge on competition when identifying trading opportunities according to a 2019 paper by Monk, Prins, and Rook. Their research shows that investors are given ample reason not only to seek access to alternative datasets but to build internal capacity for working with and acting on them.

The fields of ML and NLP are constantly developing, and new research is being produced round the clock. Recent progress in ML has been driven both by the development of new learning algorithms and theory and by the ongoing explosion in the availability of online data and low-cost computation (Jordan, 2015). In the past several years a dramatic shift has occurred in the NLP research sector, moving away from traditional symbolic methods too hybrid methods incorporating new empirical corpus-based methods and the use of probabilistic and information theory, fuelled by the growth in ML research and interest.

Similar research focusing on the impact of sentiment of security prices have varying conclusions on the significance of its impact and the validity of sentiment analysis as an alternative data source for technical analysis is a divisive topic. Ni, Su, Wang, and Ying found, in their research into a stock evaluation index based on public opinion analysis (2019), that there was no significant correlation between the general sentiment of individual investors and the evaluation of consulting institutions. Alternatively, a 2019 study by Henrikson and Hultberg into public sentiment on twitter and stock performance found that if the data quality is good enough a high correlation between the public opinion and stock price can be found.

Despite there being vast amounts of research pretraining to the impact of sentiment of security prices, financial news headlines as a data source and their impact on the U.S. markets is a relatively unexplored field.

Solutions such as BERT (Bidirectional Encoder Representations from Transformers), a language representation model designed to pre-train deep bidirectional representations from unlabelled text by jointly conditioning on both left and right context in all layers (Devlin et al. 2019), can achieve some of the best accuracies for NLP tasks. However, as a pretrained model bias is added which can lead to local minimum and suboptimal solutions, this can be further catalysed by finance specific language used within the scope of this problem.

7 Approach

As aforementioned in the introduction (see section 5), this project is bound by time and thus its scope and scalability are not limitless. Thus, it is necessary to categorise the aims and objectives of the project.

- **A1:** Complete research into the various aspects of the project.
- **A2:** Construct an accurate and complete training data set suitable for algorithmic supervision.
- **A3:** Produce a functioning sentiment analysis classification model through machine learning.
- **A4:** Conduct statistical analysis on a minimum of 5 security's including the overarching market.
- **A5:** Complete Documentation and Report.

7.1.1 Functional Requirements

The '*MoSCoW*' task prioritization method has also been implemented to manage the individual functional requirements of each aim.

A1: Complete research into the various aspects of the project.

- **SHOULD** Research techniques that can be used to implement the tool.
- **SHOULD** Research technologies that can be used to implement the tool.

A2: Construct an accurate and complete training data set suitable for algorithmic supervision.

- **MUST** construct the training set corpus.
- **MUST** conduct basic pre-processing techniques.
- **MUST** implement Named Entity Recognition for individual securities.
- **COULD** conduct a survey to determine the sentiment value of specific terms for labelling.

A3: Produce a functioning sentiment analysis classification model through machine learning.

- **MUST** produce a sentiment classification tool.
- **MUST** implement a machine learning algorithm.
- **SHOULD** test for an optimal machine learning algorithm.
- **COULD** optimise a machine learning model through hyper parameter tuning.
- **COULD** account for Amplifiers, De-Amplifiers, Negators and Adversative Conjunction.

A4: Conduct statistical analysis on a minimum of 5 security's including the overarching market.

- **MUST** plot the price and sentiment of individual securities on a time series.
- **MUST** calculate the correlation coefficient for the price against sentiment.

- **SHOULD** plot the scatter graph for the price against sentiment with a linear regression line.

A5: Complete Documentation and Report.

- **MUST** write a detailed final report describes all aspects of the project in detail along with supporting information.
- **SHOULD** produce legible, understandable, and maintainable code documentation.
- **SHOULD** provide a justification and evaluation of the produced results.

7.1 Data Collection

7.1.1 Data Sources

Financial News Headlines - Gathered from the following Kaggle dataset: https://www.kaggle.com/notlucasp/financial-news-headlines. Within this set, data has been scraped from CNBC, the Guardian, and Reuters official websites. Each dataset contains the headline, the last updated date and CNBC and Reuters contain preview text of the articles. The data is stored in 3 separate csv files. The data contains raw text strings and datetime values.

Securities Data - Stock data will be collected from a 'Yahoo Finance' API, documentation for which can be found here: https://pypi.org/project/yfinance/. The data selected from the API will include close prices of the stocks during the analysed period and will exported to a csv file for storage. The data countians numeric non-integer values and datetime values.

These sources have been chosen as they provide data necessary to investigate the hypothesis *'the sentiment of financial news headlines reflects and directs the performance of the U.S. stock market'*. They contain a high value and veracity as well as a large enough volume to train an accurate model. Additionally, they have no velocity and a variety high enough to negate any inaccuracy's but not too large to add unnecessary complications.

7.1.2 Data Cleaning

Data cleaning is the process of deleting and correcting corrupt or inaccurate values from a dataset, and refers too identifying incomplete, inaccurate, incorrect, or irrelevant parts of the data and then replacing, modifying, or deleting the dirt or coarse data. The data cleaning process for the data sources within this project can be broken down into four stages:

- **Duplicate Removal** – Removing any duplicate records from within the dataset.

- **Missing Data Removal** - Any records containing a missing value will be removed from the dataset entirely.

- **Structural Errors Correction** – Placing a consistent structure of the dataset.

- **Data Type Correction** – All values of the same type require a consistent data type and structure.'

- **Ordering** – Records are stored in order of the date.

The data cleaning stage is necessary to improve overall data quality and in doing so increase overall productivity.

7.1.3 Data Aggregation

Once the data has been cleaned and the format, structure and type of data is consistent across all 3 datasets, data aggregation will occur, and the datasets will be combined and compiled into one large dataset for ease of use.

7.1.4 Pre-Processing

Text data is rich in content but highly unstructured, as features are not explicitly available in text data, a process Is required to extract features from the unstructured text. One way to approach this is to consider each word as a feature and find a measure to capture whether a word exists or does not exist in a sentence. As this feature vector would be of substantial size it would suffer from the 'curse of dimensionality' (Zhang 2013) (which refers to various negative phenomena that arise when organizing and analysing data in high-dimensional spaces), so the number of words should be reduced via a series of techniques.

- **Tokenization** – Tokenization is a way of separating a piece of text into smaller units called tokens.

- **Stemming** – Stemming is the process of reducing a word to its inflectional forms, using a heuristic to remove prefixes and suffixes.

- **Lemmatization** – Lemmatization is the process of reducing a word to its inflectional forms and derivationally related forms, using morphological analysis.

- **Stop-word Removal** – Stop word removal involves removing words that occur commonly across all documents in the corpus.

- **TF-IDF** (Term Frequency-Inverse Document Frequency) – TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents.

In addition to these basic concepts advanced linguistic annotation can be implemented through the application of syntactic and grammatical rules to identify the boundary of a sentence or term despite ambiguous punctuation.

- **N-grams** – The inclusion of *n* individual groupings of words i.e., 'New York'

- **POS Tagging (Part of Speech)** – Identifies a words role and relationship within a sentence.

- **Dependency Parsing** – Identifies hierarchical relationships among words.

Pseudocode:

```
FUNCTION preProcessing(text: String):
    text = tokenization(text)
    text = text - stopwords
    text = text ∩ dictionairy
    FOR word IN text:
        word = lemmatize(word)
        word = stemming(word)
    RETURN text
```

7.1.5 Subjective Vs Objective

Headlines may be either objective or subjective in nature. Objective headlines will contain no sentiment as they are not influenced by an individual's feelings and are considered to represent fact. Subjective headlines are based or influenced by a person's feelings therefore will contain sentiment.

Objective Headline: "The S&P500 is remaining stable"

Subjective Headline: "Google stock has a positive outlook for the future"

As objective headlines will provide no insight into the writer's opinion, prior to training the ML classifier they will be deemed as 'neutral' in annotation.

7.1.6 Scope

This study will evaluate financial news headlines from December 2017 to July 2019 and will focus on a limited selection of US stocks as well as the S&P 500 index, this should provide a fair overview of the market through both 'bearish' and 'bullish' periods and demonstrate the impact of sentiment on the market. All decisions in scope have been made due to the limited time interval and all available data.

7.2 Sentiment Analysis

7.2.1 Method

Sentiment analysis can be split broadly into two separate strains: 'polarity' and 'beyond polarity'. Where 'polarity' analysis only focuses on the positive, negative, or neutral sentiment within a piece of text, 'beyond polarity' analysis looks at different emotional states. This study will focus on polarity sentiment classification as the market can only fluctuate in two directions and the impact of individual emotion states would have a negligible impact on price movements.

ML based sentiment analysis will be used in this project. A ML model is trained to classify sentiment based on both the word and their order. This method consists of three stages: Feature extraction, Training and Prediction. The success of this method is highly dependent on the classification algorithm implemented so various algorithms will be tested against different evaluation measures. The chosen algorithm will then be trained and tweaked for optimality.

Sentiment Classification Flow Chart:

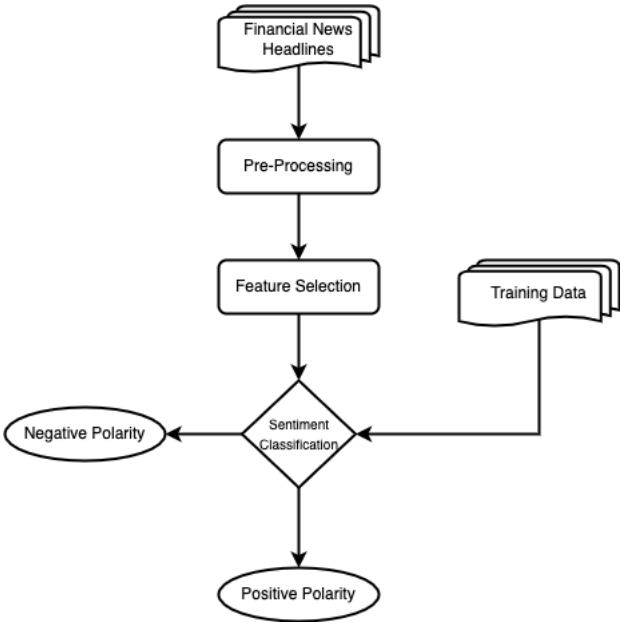

Figure 1: The sentiment classification flow chart.

7.2.2 Bag-Of-Words Features

Many ML models cannot be given training data directly, it first must be transformed into a feature vector (an *n* dimensional vector which acts as a numerical representation of the data) to be processed. A key challenge consists of converting text into a numerical value without it losing its meaning. Reduction of the 'Dimensionality' of features in a dataset is necessary as ML algorithms can suffer under a vast number of observations, leading to a reduced ability to train an effective model.

Within sentiment analysis a 'Bag-Of-Words' representation (BoW) is used to perform feature extraction. The BoW model is a simplifying representation where text is represented as the bag of its words, disregarding grammar and word order but keeping multiplicity.

A variance threshold will be applied to the features to reduce the dimensionality and disregard features with negligible change between observations to select a subset of significant features from the entire pool. The key trade-off at this step is the choice of a larger vocabulary to better reflect the text source at the expense of more features and a higher model complexity.

From the utilization of BoW features a ‘Document-Term Matrix’ can be computed in a recursive process for each headline. The ‘Document-Term Matrix’ is a crude simplification because it abstracts from word order and grammatical relationships. Nonetheless, it often achieves a high performance in sentiment classification given a limited time frame.

Document-Term Matrix Example:

Document	Term 1	Term 2	…	Term n
Headline 1	0	2	1	0
Headline 2	3	0	1	1
Headline …	1	0	2	0
Headline n	0	1	0	2

7.2.3 Named Entity Recognition

Named Entity Recognition (NER) is the problem of locating and categorizing important nouns and proper nouns in a text (Mohit 2014). NER will be performed to locate and classify specific named entities within the unstructured text and classify them into pre-defined categories. Entities can be names of people, locations, times, quantities, monetary values, percentages and more. In the context of this study company names can be categorized as organizations, thus headlines impacting specific organizations can then be identified and the organizations name can be replaced in the text to reduce overfitting within the dataset towards a particular company name.

7.3 Machine Learning

7.3.1 Machine Learning Process

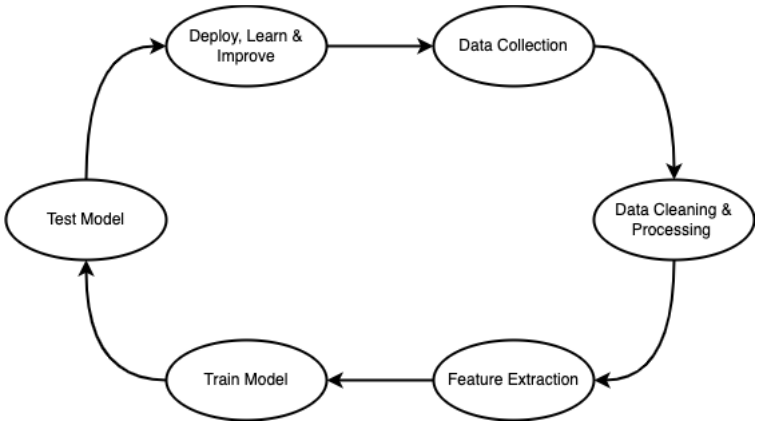

Figure 2: The machine learning process.

Data Collection – The dataset can be populated by data from various sources. The more diverse and extensive the data used is, the higher accuracy the ML model's findings would be.

Data Cleaning & Processing – Involves cleaning and aggregating all data used within the model to transfer it into a more desirable format. Basic analysis pre-processing steps then occur to reduce dimensionality in preparation for feature extraction.

Feature Extraction – The process of transforming raw data into numerical features that can be processed while preserving the information within the original dataset.

Train Model – Using the training data, a model will be constructed as defined by the algorithm implemented, to identify patterns.

Test Model – Using the testing data, the model will make predictions and its ability will be addressed against a series of evaluation metrics.

Deploy, Learn & Improve – This process entails experimentation with multiple algorithms, optimising, evaluating, and refining to discover the most suitable fit for this specific problem.

7.3.2 Supervised vs Unsupervised

Supervised learning is a ML approach that is defined by its use of labelled datasets. These datasets are designed to train or 'supervise' algorithms into classifying data or predicting outcomes accurately. Labelled input and outputs allow the model to measure its accuracy and lean overtime. Supervised learning problems include Classification and Regression.

Unsupervised learning uses ML algorithms to analyse and cluster unlabelled datasets. These algorithms discover hidden patterns in data without the need for human intervention. Unsupervised learning problems include Clustering, Association and Dimensionality reduction.

A supervised learning approach will be taken for this study, due to the problem scope.

7.3.3 Classification vs Regression

Classification algorithm finds a function that helps in dividing the dataset into classes based on different parameters. The task of a classification algorithm is to find the mapping function that maps the input (x) to a discrete output (y).

Regression algorithm finds the correlations between dependent and independent variables. The task of a regression algorithm is to find the mapping function that maps the input variable (x) to the continuous output variable (y).

A classification approach will be taken for this study as sentiment polarity is being considered rather than a sentiment based numeric value.

7.3.4 Cross Validation (CV)

Cross validation is a resampling method that uses different portions of the data to test and train a model on different iterations. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

The models can be compared to test for overfitting of the data. CV is highly effective when working with smaller datasets, or when the distribution of the dataset is skewed.

CV Advantages:

- Utilize all training data.
- Gaining additional metrics.
- Model Stacking.
- Working with dependent/grouped data.
- Parameter fine tuning.

K-Fold CV vs Monte Carlo CV:

K-Fold Technique:

1. Split the training data into K equal parts
2. Fit the model on k-1 parts and calculate test error using the fitted model on the k^{th} part.
3. Repeat k times, using each data subset as the test set once.

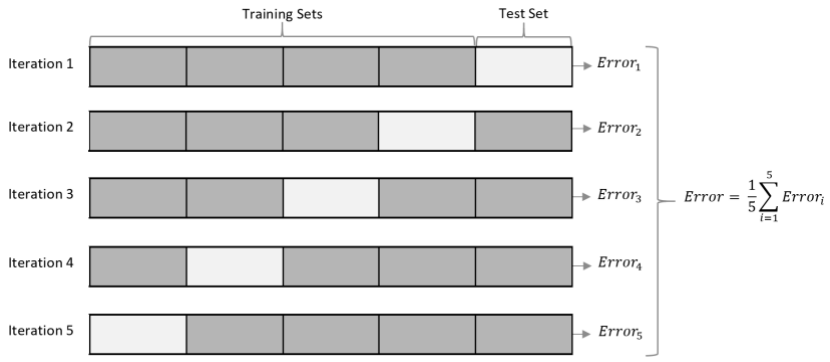

Figure 3: The process of K-Fold CV (Patro 2021).

Monte Carlo Technique:

1. Split the training data randomly (for each iteration, the train-test split is different).
2. Fit the model on train data set for that iteration and calculate test error using the fitted model on test data.
3. Repeat many iterations and take the average of the test errors.

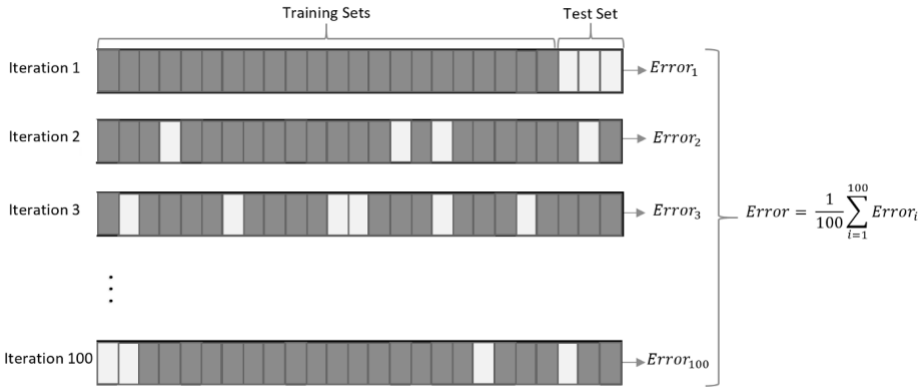

Figure 4: The process of Monte Carlo CV (Patro 2021).

K-Fold CV will be implemented in this study as each dataset is tested exactly once (allowing all the training data to be utilised) and the results are unbiased.

7.3.5 Standardisation

Standardization, a common requirement for many ML classifiers, is a scaling technique where the values are cantered around the mean with a unit standard deviation (gaussian with zero mean and unit variance). Standardization is necessary when features of an input dataset have large differences between their values. The difference in these ranges of initial features causes trouble to many ML models, one of these being the Support Vector Machine.

Support vector machines tries to maximize the distance between the separating plane and the support vectors. If one feature has very large values, it will dominate over other features when calculating the distance. Therefore, standardization is necessary to give all features the same influence on the distance metric.

7.3.6 Fitting

Model fitting is a measure of how well a ML model generalizes to similar data to that on which it was trained. ‘Overfitting’ occurs when a statistical model fits exactly against its training data whereas ‘Underfitting’ occurs when a statistical model doesn’t fit against its training data enough. For a predictive model to function effectively it is important that it is ‘well-fitted’ to produce more accurate outcomes.

The prediction error for any ML model can be broken down into three parts:

- **Bias Error** – Simplifying assumptions made by a model to make the target function easier to learn.

- **Variance Error** – Amount that the estimate of the target function will change if different training data was used.

- **Irreducible Error** – Cannot be reduced regardless of algorithm. Error introduced from chosen framing of the problem.

Ideally, a model would have a low variance and a low bias. However, in practice, a predictive model will have a 'Trade-Off' between the two (tension between the error introduced by the bias and the variance).

An *overfitted* model will have low bias and high variance.
An *underfitted* model will have high bias and low variance.

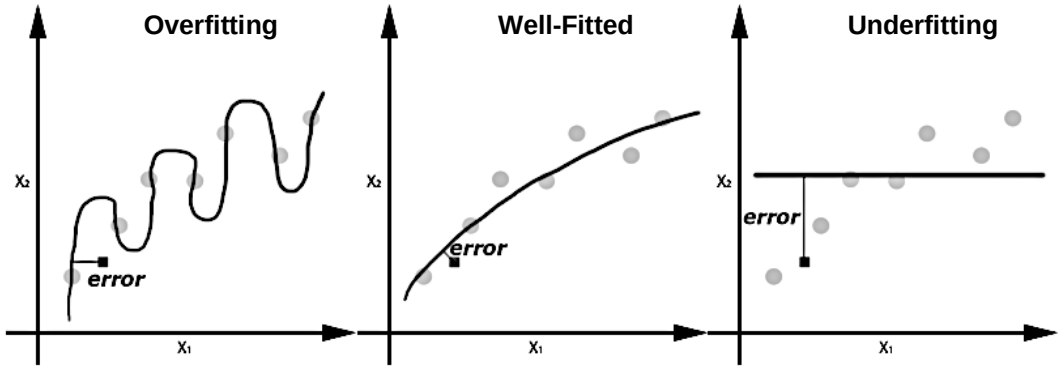

Figure 5: An underfitted, well-fitted and overfitted model (Parveez 2020).

7.3.7 Classification Algorithms Evaluation

The following classification algorithms were chosen to be evaluated due to their wide application for text classification problems, in particular sentiment analysis. These traditional models are widely used for large-scale sentiment analysis, such as the problem at hand, due to their scalability capabilities and their suitability for determining polarity.

Naïve Bayes:

Defined by Webb (2017), Naïve Bayes is a learning classification algorithm that utilizes Baye's rule together with a strong assumption that the attributes are conditionally independent of the class, hence the 'Naïve'. While this independence assumption is often violated in practice, naïve bayes nonetheless often delivers competitive classification accuracy. Coupled with its computational efficiency and many other desirable features, this leads to naïve bayes being widely applied in practice.

Bayes Rule:

$$\mathrm{P}(y \mid \mathbf{x}) = \mathrm{P}(y)\mathrm{P}(\mathbf{x} \mid y)/\mathrm{P}(\mathbf{x})$$

Advantages:

- Simple to implement and the conditional probabilities are easy to evaluate.
- High processing speed – no iterations since the probabilities can be directly computed.

Disadvantages:

- Conditional Independence assumption does not always hold. In most situations, features show some form of dependence.
- Zero probability problem: When encountering words in the test data for a particular class that are not in the training data, we might end up with zero class probabilities.

Binary Logistic Regression:

Logistic regression is a ML algorithm used for classification problems, it is a predictive analyst algorithm and based on the concept of probability and is used to obtain odds ratio in the presence of more than one explanatory variable (Sperandei 2014). A logistic regression model is like that of multiple linear regression but uses a more complex cost function to enable it to perform classification, this cost function can be defined as the 'Sigmoid function' (the response variable is binomial). The result is that the impact of each variable on the odds ratio of the observed event of interest.

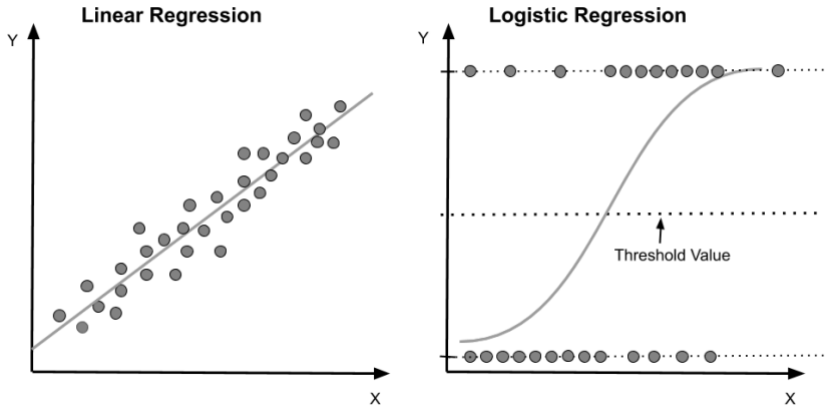

Figure 6: The difference between linear and logistic regression (Mheta, A).

Sigmoid Function:

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$

Advantages:

- Easy to implement, interrupt, and very efficient to train.
- High processing speed when classifying unknown records.
- Good accuracy for many simple data sets and performs well when the data set is linearly separable.

Disadvantages:

- Limitation due to linearity assumption between the dependent variables and the independent variables.

- Overfitting can occur if the number of observations is less than the number of features.

Support Vector Machine:

Defined by Jakkula (2006), a Support Vector Machine (SVM) is a classification and regression prediction tool that uses ML theory to maximize predictive accuracy while automatically avoiding overfitting to the data. SVM can be defined as systems which use hypothesis space of a linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory.

Advantages:

- Improved performance when there is a clear margin of separation between classes.
- Effective in high dimensional spaces.

Disadvantages:

- Not suitable for larger data sets.
- Poor performance when the data set has a considerable amount of 'noise'.
- Overfitting can occur if the number of observations is less than the number of features.

7.3.8 Testing Metrics

Confusion Matrix:

A confusion matrix produces an output describing the complete performance of the model. In a confusion matrix, expected values can be compared to predicted values. It can be utilized to visualise how a binary classification model performs:

Number of samples: n	PREDICTED: **NO**	PREDICTED: **YES**
ACTUAL: **NO**	**TN**	**FP**
ACTUAL: **YES**	**FN**	**TP**

The four measures made by the matrix are detailed as follows:

- True Positive (**TP**) - when a classifier predicts a positive and it was correct.
- False Positive (**FP**) - when a classifier predicts a positive and it was incorrect.
- True Negative (**TN**) - when a classifier predicts a negative and it was correct.
- False Negative (**FN**) - when a classifier predicts a negative and it was incorrect.

Using these, other measures of evaluation can be derived:

Classification Accuracy:

Classification Accuracy is the core measurement of how effective any ML algorithm is. It is the ratio of correct predictions to the total number of input samples. This metric will be the core measure to determine what classifier to use, however it can be very misleading in real-word scenarios due to class imbalance, thus other measures are required to evaluate continuity.

Classification Precision and Recall:

Classification Precision describes the percentage of positive predictions which the model classifies correctly.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Classification Recall describes the percentage of positive predictions that are predicted as positive.

$$\text{Recall} = \frac{TP}{TP + FN}$$

To fully evaluate the effectiveness of a model, both precision and recall must be examined. Unfortunately, precision and recall are often in tension i.e., improving one typically reduces the other.

F1 Score:

F_1 Score is the harmonic mean between precision and recall. The range for the F_1 score is [0, 1]. It describes both the precision and robustness of the classifier. The greater the F_1 score the better the performance of the model.

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

7.3.9 Hyperparameter Optimisation

Hyperparameter tuning is choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a model argument whose value is set before the learning process begins. The optimality of these hyperparameters help to perfect the bias-variance ‘Trade-Off’ and support the model in being ‘well-fitted’.

To select the optimal hyperparameters each proposed hyperparameter setting is evaluated against the model and the one that produces the best performance is

chosen. The two main optimization algorithms are ‘Random search’ and ‘Grid Search’.

Random Search consists of defining a search space as a bounded domain of hyperparameter values and randomly sample points within that domain.

Grid Search consists of defining a search space as a grid of hyperparameter values and evaluating every position in the grid.

A randomized search strategy will be implemented, if necessary, for this problem as it superior for discovery and finding hyperparameter combinations that would not have been guessed intuitively.

Within the scope of this study, Naïve Bayes and Logistic Regression have no critical hyperparameters to tune, as they will have a negligible impact on the performance of the model. SVM’s core hyperparameter is the ‘Kernel’, the kernel is responsible for mapping the observations into some feature space, ideally making the observations more linearly separable after this transformation.

7.3.10 Full Pipeline

Conducting classification on the financial news headlines will require a ML pipeline to bring all aspects of the classification process together. An ML pipeline is and end-end construction that orchestrates the flow of data into, and output from, a ML model. Under the constraints of this project the exposed static structure of the classification pipeline can be represented by the following UML (Unified Modelling Language) diagram.

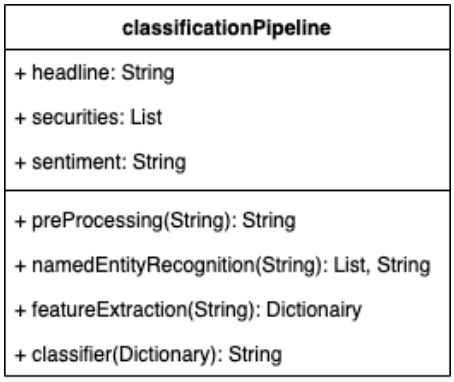

Figure 7: Classification Pipeline UML

7.4 Data Analysis

7.4.1 Selected Securities

SPY (S&P 500) – The Standard and Poor’s 500 is a stock market index tracking the performance 500 large companies listed on stock exchanges in the United States. It is one of the most followed equity indices.

To ensure a varied set of companies from differing industries, the largest company by market capitalization of the largest sectors within the S&P 500 were chosen. Larger corporations were chosen as they would be more impactful and were likely to have more headlines targeted towards them. Various industries were chosen so the sentiments impact could be monitored and measured over differing sectors.

Securities under investigation:

- **GOOGL** (Google) – Google LLC is an American multinational technology company that specializes in internet related services and products, which include a search engine, online advertising technologies, cloud computing, software, and hardware.

- **FB** (Meta) – (*named 'Facebook' during the period of headlines being analysed*) Meta Platforms, Inc. is an American multinational technology conglomerate. The company is the parent organization of Facebook, Instagram, and WhatsApp among other subsidiaries.

- **AMZN** (Amazon) – Amazon.com, Inc. is an American multinational technology company which focusses on e-commerce, cloud computing, digital streaming, and artificial intelligence.

- **TSLA** (Tesla) – Tesla Inc. is an American electric vehicle and clean energy company. Tesla designs and manufactures electric cars, battery energy storage from home to grid-scale, solar panels and solar roof tiles, and related products and services.

- **APPL** (Apple) – Apple Inc. is an American multinational technology company that specializes in consumer electronics, software, and online services.

- **V** (Visa) – Visa Inc. is an American multinational financial services corporation that facilitates electronic fund transfers throughout the world.

- **JNJ** (Johnson and Johnson) – Johnson & Johnson is an American multinational corporation that develops medical devices, pharmaceuticals, and consumer packaged goods.

- **WMT** (Walmart) – Walmart Inc. is an American multinational retail corporation that operates a chain of hypermarkets, discount department stores, and grocery stores from the United States.

- **XOM** (ExxonMobil) – ExxonMobil is an American multinational oil and gas corporation.

7.4.1 Hypothesis Test

Correlation coefficients are indicators of the strength of the linear relationship between two different variables, x, and y. This study will utilize the Pearson product-moment correlation, to measure the linear relationship between headline sentiment

and security price movement. If the coefficient is greater than a specific threshold, the relationship between the two will be deemed statistically significant.

Correlation Coefficient Equation:

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{\left[n\sum x^2 - (\sum x)^2\right]\left[n\sum y^2 - (\sum y)^2\right]}}$$

The correlation coefficient for each will be calculated through the gradient of a linear regression line. This calculation works as correlation quantifies the strength of a linear relationship between a pair of variables. Thus, the gradient of a linear regression line directly demonstrates the strength of the variable's relationship and therefore its correlation.

Two null hypotheses will be tested for each aspect of the core hypothesis that this study considers; one hypothesis to test the market price and sentiments 'reflection' and one for its 'direction'.

Direction H_0 Null Hypothesis: The sentiment of financial news headlines does not direct the performance of a U.S. security.

Reflection H_0 Null Hypothesis: The sentiment of financial news headlines does not reflect the performance of a U.S. security.

A significance level of 0.05 will be employed due to the amount of data present and the problem under consideration.

Significance Level: $\alpha = 0.05$ (5%)

The t-value and p-value will then be calculated to determine if the correlation is repeatable and weather it meets the significance level. The decision can then be made weather to accept or reject the null hypothesis.

$$t = \frac{r\sqrt{n-2}}{\sqrt{1-r^2}}$$

7.5 Data Visualisation

7.5.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations as defined by Patil (2018).

EDA techniques will be used throughout the project to adjust where suitable considering what is discovered. Graphics will mainly be produced to assess the quality and quantity of the data sets. These can aid decisions later in the process as to how to approach specific tasks. Additional NLP specific EDA techniques will also be used however, to provide insight into what the data consists of and how individual headlines are structured and the variety of words they consist of.

NLP EDA Techniques:

- **POS Tagging and Frequency Analysis** - Analysis of the types of words and their respective frequency within the data set provides insight into the structure of individual headlines.

- **Word Cloud** - Provides a visual representation of text data, which can be used to depict key words within a vast amount of data. Key words are typically by themselves and depicted in importance by either size, font, or colour.

- **Concordance** – Allows for the search of words or phrases and their given context in parallel to the headline.

7.5.2 Model Diagnostic Visualisations

Three types of diagnostic tool will be implemented to assess the model and diagnose potential issues. In the context of this study the diagnostic visualisations can be generated through the cross-validation process to evaluate the model to assess how it can be further adapted or modified to improve performance.

Firstly, a learning curve will be plotted. A 'Learning Curve' is a plot of model learning performance over experience or time. Learning curves are a widely used diagnostic tool in ML for algorithms that learn from a training set incrementally. Assessing the learning curves of models can be used to diagnose problems with learning such as the fitting or whether the training and validation sets are representative. Within this study a 'Performance' learning curve (calculated on the metric by which the parameters of the model are being optimized) will be generated.

Learning Curve: Line plot of learning (y-axis) over experience (x-axis).
Train Learning Curve: Learning curve calculated from the training dataset that gives an insight into how well the model is learning.
Validation Learning Curve: Learning curve calculated from a hold-out validation dataset that gives an idea of how well the model is generalizing.

Secondly, a model scalability graph will be plotted. This demonstrates the times required by the model to train various sizes of training dataset.

Thirdly, a model performance graph will be plotted. This demonstrates how much time is required to train the model for each training size.

7.5.3 Time Series Graph

The production of the time series will be an attempt to analyse how the prevailing sentiment reflects the market. A Time Series is a series of data points graphed in time order at successive equally spaced points. In the financial space, a time series tracks the movement of the chosen data points, such as a securities price, over a specified period with data points recorded at regular intervals. No restrictions exist on the period that must be included allowing data to be gathered in a method that provides value to an investor or analyst. In the context of this project the stock price will be plotted on a time series along with the sentiment polarity under the time frame outlined by the scope of this project.

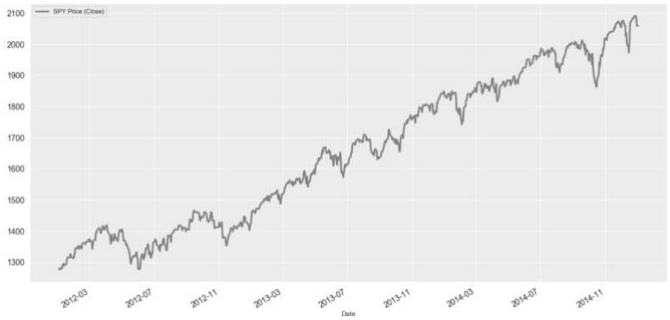

Figure 8: A Time Series Plot (S&P 500 from 2012-03 to 2012-11) (Travora 2019).

7.5.4 Scatter Plot Graph

The production of the scatter plot will be an attempt to analyse how the prevailing sentiment directs the market. A scatter plot uses multiple data points to represent values for two different numeric values, the position of each dot on the horizontal and vertical axes indicates the data points value. Scatter plots are used to observe relationships between variables, in the context of this project a linear regression line will be plotted to fit a model between the two variables, through this line the correlation can be calculated through studying the gradient. The numeric values used will be the change in sentiment and the change in price.

7.6 Technologies and Techniques

7.6.1 Language

Python programming language is an open source, interpreted, high level language and provides a great approach for object-oriented programming. It is one of the leading languages for data science. Python provides leading functionality to deal with mathematics, statistics, and scientific function as well as a vast number of libraries to deal with data science application.

The ’Simple Concordance Program’ (SCP), developed by Alan Reed (2007), will be used for a section of the EDA conducted.

7.6.2 Development Environment

‘Jupyter’ is an open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text, and multimedia resources into a singular notebook document.

Notebooks are well suited for data science projects as they allow a researcher to execute code, monitor results, make modifications, and repeat in an iterative process between the user and the data.

7.6.3 Core Libraries

To implement what has been discussed a series of external python libraries will be required for NLP, ML, and data processing. The main libraries that will be used are detailed as follows:

- **Scikit-Learn** – A python ML library that features various classification, regression, and clustering algorithms. Within this project the module will be used for building, training, and testing ML classifiers.

- **Matplotlib** – A python plotting library. This module will be used for data visitation preposes, both exploratory and conclusively.

- **NLTK** (Natural Language Tool Kit) – A python library for symbolic and statistical NLP for English. This will provide data processing and cleaning functionality to the project.

- **Pandas** – A python library for data manipulation and analysis. Offering data structures and for manipulating numeric tables and time series. The Pandas 'Data Frame' will be the main storage data structure used throughout, for both input and output of data.

- **yFinance –** An API that offers a threaded and pythonic way to download market data from Yahoo Finance. The API will provide all the security analytics necessary for computation.

- **NumPy** – A python mathematical library that provides a large collection of high-level mathematical functions as well as support for large, multi-dimensional arrays and matrices. These mathematical functions and arrays will be utilized throughout the project.

- **SciPy** – A python scientific library that provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics, and many other cases of problems. These scientific functions will be utilized when conducting statistical analysis.

- **Spacy** – A python library for advanced NLP. Offers NER and POS tagging. The libraries NER functionality will be its core point of use, allowing named entities to be replaced with their corresponding label.

8 Implementation

8.1 Data Collection and Cleaning

At the outset of the implementation of this research the data sets being analysed are to be imported so they can be modified and prepared for classification. To provide access for Jupyter the 'Pandas' python module was used to directly access the CSV files in which the headline data was stored (using the directory path for access). Each data set is then individually stored as a Pandas Data Frame variable (a two-dimensional, size-mutable, potentially heterogenous tabular data structure).

```
"""Collect Headline Data"""

import pandas

cnbcData = pandas.read_csv('Data/Original/cnbc_headlines.csv')
guardianData = pandas.read_csv('Data/Original/guardian_headlines.csv')
reutersData = pandas.read_csv('Data/Original/reuters_headlines.csv')
```

Security data is gathered through the 'yFinance' API through the implementation of a custom function to return the daily close price (list) in the specified period of the project.

```
"""Collect and Clean Market Data"""

import yfinance

def getMarketData(ticker):
    securityData = yfinance.download(ticker, start='2017-12-22', end='2020-07-18')
    securityData = securityData['Close']
    return securityData
```

Cleaning the data sets involves removing duplicates and missing values to prevent encountering any errors regarding the quality of the data in the future. In addition to just cleaning, as three separate data sets are being used to gain perspective, accompanying this are various challenges with differences in format, structure, and type ranging across the data sets. It is necessary to correct this variance to allow the data to be concatenated into a singular uniform ordered combined dataset.

To perform the basic cleaning process a short function is built that removed rows with missing values as well as duplicate values. Additionally, it resets the index of the data frame and removes any unnecessary data for the project if it is present.

All the CNBC data has an unusual datetime naming convention in which a variety of differing abbreviations are used for different months, with no consistency throughout the dataset. To account for this an additional short function is built to parse the datetime text and manually change the naming conventions into variations that can be directly converted into the 'datetime64' format. Furthermore, the 'time' aspect of the datetime data is removed (unnecessary level of detail) and the column is renamed to 'Date'. Similarly, Guardian data has the datetime set as a string, so the dates are iteratively formatted to 'datetime64' as a list based off the date style. The column name is also changed to 'Date'. In contrast, Reuters data simply requires a conversion to 'datetime64' and change column name

```
#CNBC

cnbcData = cleanData(cnbcData)
dateFormat = '%I:%M  %p ET %a, %d %b %Y'
dates = []
for item in cnbcData.iloc[:, 1].values:
    item = dateConversion(item)
    dates.append(datetime.strptime(item, dateFormat).strftime("%m-%d-%Y"))
cnbcData["Time"] = dates
cnbcData["Time"] = cnbcData["Time"].astype("datetime64")
cnbcData.rename(columns={"Time":"Date"}, inplace = True)

#Guardian

guardianData = cleanData(guardianData)
guardianData["Time"] = pandas.to_datetime(guardianData["Time"], errors = 'coerce', format="%d-%b-%y")
guardianData.rename(columns={"Time":"Date"}, inplace = True)

#Reuters

reutersData = cleanData(reutersData)
reutersData["Time"] = reutersData["Time"].astype("datetime64")
reutersData.rename(columns={"Time":"Date"}, inplace = True)
```

To produce the final combined dataset the three cleaned datasets, now in a uniform structure, are concatenated together and ordered by date. The final data frame contains 53118 unique financial news headlines along with the corresponding date that they were published.

```
"""Combine Headline Data"""

dataSets = [cnbcData, guardianData, reutersData]
headlineData = pandas.concat(dataSets)
headlineData = headlineData.sort_values(by="Date")
headlineData = cleanData(headlineData)
headlineData.info()
headlineData.to_csv('Data/all_headlines.csv')
```

8.2 Pre-Processing

8.2.1 Fundamental Techniques

The goal of pre-processing is to improve the generalizability of the final model as real-world data is often imperfect, and can contain missing data, useless data, or other unnecessary data. Removing this noise leads to the accuracy of the final model improving. To improve this generalizability, a series of steps are taken to remove and format different words. These steps all aim to reduce the number of words to leave the function down to those that are most informative for ML classification.

A custom function is implemented to allow raw text to be processed throughout the text, when necessary, either as a precursor to feature extraction, construction of the BoW model or for continuity purposes. The produced function takes advantage of a variety of techniques.

1) Tokenization - Converts raw text string, into an individual list of words. This allows the words to be individually processed in the later steps.

2) Formatting Text - Removal of all numbers, symbols (such as punctuation), and whitespace also converts all characters to lower space.

3) Stop Word Removal - Removes words that occur commonly across all documents and hence are poor discriminators.

4) Out-Of-Vocabulary Removal - Removes words that are not present in the dictionary.

5) Lemmatization - Reduces a word to its inflectional forms and derivationally related forms, using morphological analysis.

6) Stemming - Reduces a word to its stem (a proxy for a root), using a heuristic to remove prefixes and suffixes.

The ordering of steps during this process is crucial as steps are dependent on each other for processing to occur and for optimal results. Lemmatization is conducted prior to stemming to normalise irregular words, for example, torn and tearing have different stems, but the same lemma (tear).

```
"""Pre-Processing"""

from nltk.corpus import words, stopwords
from nltk.stem import WordNetLemmatizer, PorterStemmer

lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()

#nltk.download()

def preProcessing(text):
    text = namedEntityRecognition(text)[0] #Named Entity Recognition
    text = re.sub(r'[^a-zA-Z]', " ", text.lower()).split() #Formatting Words
    text = list(set(text) - set(str(stopwords))) #Stop Word Removal
    text = list(set(text) & set(words.words())) #Non-Word Removal
    for word in text:
        word = lemmatizer.lemmatize(word) #Lemmatization
        word = stemmer.stem(str(word)) #Stemming
    return text
```

A variety of errors emerged when implementing the pre-processing methods and multiple debugging sessions were required to ensure the function processed efficiently and robustly.

8.2.3 Named Entity Recognition

Performing NER on the financial news headlines is necessary to identify individual securities that the headline references (for subsequent analysis and visualisation) as well as noise reduction from the dataset through entity removal, to prevent fitting towards specific entities. To implement this a two-step approach is taken, Initial security identification and replacement and then general entity identification and replacement. This procedure is built within a function returning the securities identified from a document (list) and the modified text (string), this allows the headlines to be processed iteratively.

Initial security Identification aims to identify the securities being analysed within the raw headline and list them in a string. The words within the headline are reduced to their stem to allow entities represented with a noun or pronoun suffix such as “Microsoft’s” to be detected. The securities are then replaced in the text with a placeholder to mitigate overfitting towards specific securities. This provides full data

retention for analysis at later stages in the pipeline as well as removing the noise in the data that may have been missed by a general-purpose NER tool.

General entity Identification occurs to eliminate any remaining entities within the raw headlines for noise reduction and to replace them with their standard entity label (using the Spacey NER tool). This secondary stage compliments the first for wider usage that couldn't be achieved manually, but with lower identification accuracy.

```
"""Named Entity Recognition"""

import spacy
from spacy import displacy

NER = spacy.load("en_core_web_sm")

def namedEntityRecognition(text):
    selectedSecurities = {'S&P': 'SPY', 'google': 'GOOGL', 'amazon': 'AMZN', 'apple': 'AAPL',
                          'microsoft': 'MSFT','visa': 'V', 'johnson': 'JNJ', 'walmart': 'WMT',
                          'exxon': 'XOM', 'FB': 'facebook', 'TSLA': 'tesla'}
    securityNames = selectedSecurities.keys()
    text = text.lower().split()
    for word in text:
        word = stemmer.stem(str(word)) #Reduce Company To Stem
    securities = []
    for security in securityNames:
        if security in text:
            text[text.index(security)] = 'COMPANY' #Company Name Removal
            securities.append(selectedSecurities[security]) #Identify for Analysis
    if len(securities) == 0:
        securities.append('^GSPC')
    text = ' '.join(text)
    text = NER(text)
    text = ' '.join([t.text if not t.ent_type_ else t.ent_type_ for t in text]).lower()
    return text, securities
```

8.3 Exploratory Data Analysis

8.3.1 Distribution

Investigating the distribution of both data sources is conducted to ensure there are no impactful frequency discrepancies throughout the data that could have a meaningful effect on the project at later stages. To conduct this testing a time series of the price of the S&P500 is plotted in USD, to examine the frequency at which the closing price was recorded. Additionally, a bar chart is plotted recording the number of headlines within the data set per month. Both plots range the full scope of the study.

Figure 9: Time Series (Price of the S&P 500 in the scope of the project).

The time series visualisation produced infers that the close price is not reordered each day as the US markets are not always open, they run through Monday to Friday (7am-7pm).

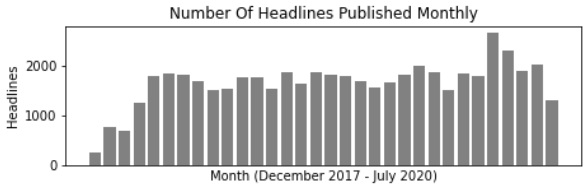

Figure 10: Bar Chart (Frequency distribution of headlines in the datasets).

The bar chart visualisation displayed that there is a lack of headlines for the months of December 2017 and January 2018 (<100). In contrast there is an estimated average of around 1000 headlines per month for remainder of the time frame.

These findings are considered when evaluating results as insufficient data could cause errors when attempting to plot visualisations (no closing price on non-trading days) later or the findings could be skewed due to an insufficient sample size (lack of headlines in the earliest two months).

8.3.2 Word Cloud

'Word Clouds' provide a visual representation of text data, which can be used to depict key words within a vast amount of data. Key words are typically by themselves and depicted in importance by either size, font, or colour. A 'Word Cloud' is generated for each headline dataset (CNBC, Guardian, and Reuters), to explore the differences and similarities in the headlines written. Every Individual word for every headline in every dataset is joined to a unique list, the 'stop words' are removed from the lists and a corresponding 'Word Cloud' is generated.

Figure 11: Word Clouds for each data set being evaluated.

It appears the individual Word Clouds all have a unique central focus. Reuters focuses on a US centric financial discussion, well suited for the study being conducted. The guardian is more UK centric, understandably as the guardian is a British publication. Alternatively, CNBC tends to focus on global financial news.

No action is required to be taken based of these exploratory results as the strong US-UK trade relationship means there is a direct correlation between the markets. Thus, sentiment polarity in the UK markets should still have an impact on the US economy, under the assumption that this studies hypothesis holds true.

8.3.3 Word Type and Frequency

Analysis of the types of words and their respective frequency within the data set provides insight into the structure of individual headlines and what can be derived from them. To produce a visual representation of this a singular string containing all the raw text headlines is compiled and basic pre-processing stages such as tokenization and text formatting are performed. POS (part-of-speech) tagging is

applied to the string to assign a category to each individual word based on both its definition and context. The frequency of occurrence of each word type can then be counted and represented as a bar chart.

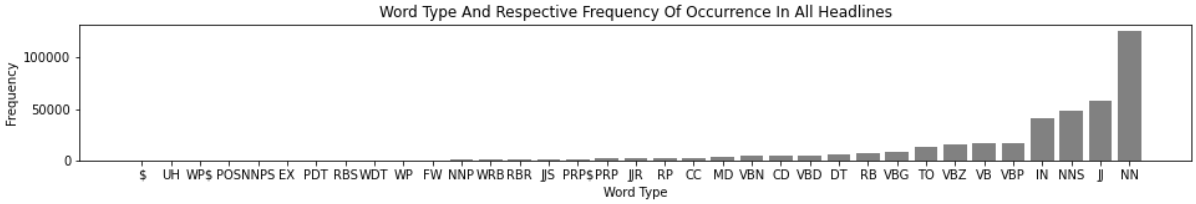

Figure 12: A Bar Chart (Frequency of individual word types in the datasets).

The tags with the highest frequency are as follows:

- **NN** – Noun, common, singular, or mass (185610)
- **JJ** – Adjective or numeral, ordinal (82931)
- **NNS** – Noun, common, plural (69855)
- **IN** – Preposition or conjunction, subordinating (63558)
- **VBP** – Verb, present tense, not 3rd person singular (25413)

These results are considered when selecting a suitable ML algorithm for classification as different classifiers calculate interactions and singularity of words through differing methods.

8.3.4 Simple Concordance Program

The SCP is utilised at various stages of the project to investigate whether the code is functioning correctly and making the exact adjustments to the text that were intended. Producing concordances provides an easy method to the context of specific words/ phrases. It is utilised post NER to ensure specific named entities have been removed. Additionally, the SCP allows for the specification of words using a prefix or suffix and the production of word lists and the exclusion of stop words via a '*stoplist*', this was utilised when constructing the pre-processing function to ensure its individual aspects were functioning as expected.

8.4 Training Dataset Generation

8.4.1 BoW Generation

The BoW representation outlines the Document Term-Matrix ready for feature extraction as well as outlining the dimensionality going into to the classifier. The list of words initially begins empty, generating the BoW involves iteratively processing each headline through the pre-processing function described previously and then adding them to the list of words if they are not already present and unique.

TF-IDF is deemed unnecessary for this specific sentiment analysis task. Aggressive feature selection tends to result in a loss of information and the assumption that some features are 'irrelevant' may bias the classifiers. There is clear tension present between dimensionality reduction and information retention, and it is felt that the number of features was reduced with enough significance (55.64% reduction)

through the standard pre-processing procedure and that TF-IDF would not be required to further reduce this value.

```
"""Generate Word Set (Bag-Of-Words Representation)"""

import numpy
import time

wordSet = []
start = time.time()
for headline in headlineData["Headlines"].to_list():
    composedHeadline = preProcessing(headline)
    wordSet = numpy.union1d(wordSet, composedHeadline)
end = time.time()
textfile = open("Data/wordset.txt", "w")
for element in wordSet:
    textfile.write(element + "\n")
textfile.close()

print("Runtime: " + str(round((end - start), 2)))
```

8.4.2 Feature Extraction

To perform classification features will need to be extracted from the headlines and passed into the ML classifier. Executing this the individual occurrence of words within a pre-processed headline are counted. These count values are appended to their relevant counterparts in a dictionary with keys corresponding to the BoW model as aforementioned. This functionality is built into a function returning the dictionary and taking the BoW (list) and document (list), allowing it to be performed iteratively over a series of headlines within the final classification pipeline.

```
"""Feature Extraction"""

def featureExtraction(words, document):
    wordFrequency = dict.fromkeys(words, 0)
    for word in document:
        if word in words:
            wordFrequency[word] = document.count(word) #Frequncy Of Occurrence
    return wordFrequency
```

8.4.3 Manual Annotation

Initially, it was intended, to produce a training data corpus, that a sector of the data set would be allocated to training and the sentiment polarity would be manually annotated based on the human annotators view of the headline. This would require multiple participants to share the workload and to mitigate any potential bias within the labelling, which in turn would require ethical consideration.
Conducting annotation manually is highly time consuming and error prone. To improve the labelling rate of the process an interactive python script is written allowing a random label to be selected, manually annotated, then added to the training data and removed from the larger dataset.

```
#Interactive Annotation Function
def annotateTrainingData():
    #Instructions
    print("Evaluate the overall sentiment of each headline:")
    print("If positive enter 'positive', if negative enter 'negative' and if nuetral enter 'nuetral'.")
    print("Enter 'END' at any point to stop the process!")
    unprocessedHeadlines = headlineData["Headlines"].to_list()
    for headline in unprocessedHeadlines:
        headline = namedEntityRecognition(headline)[0]
    entry = ""
    while entry != "END":
        headlineIndex = random.randint(0, len(unprocessedHeadlines)) #Random headline Selection
        headline = unprocessedHeadlines[headlineIndex]
        unprocessedHeadlines.remove(headline)
        entry = input(headline + " - Enter the sentiment of this headline: ") #Labeling
        if entry != 'END':
            #Process Dataset For Machine Learning
            sentiment = entry
            features = list(featureExtraction(wordSet, preProcessing(headline)).values())
            entry = [headline, sentiment]
            for feature in features:
                entry.append(feature)
            #Add Entry To Data Frame
            annotatedTrainingData.loc[len(annotatedTrainingData)] = entry
            annotatedTrainingData.to_csv('Data/Training/annotated_training_data.csv')
            headlineData.drop(headlineIndex, axis=0)
            headlineData.to_csv('Data/all_headlines.csv')
```

This would be the optimal method for accuracy however as annotation can be controlled for accuracy and any potential bias can be mitigated. Additionally, the training data is taken directly from the source data set so the observations made will share a higher level of similarity with the headlines being fed into the classifier.

The issue with this method is the time consumption, 1 headline can be annotated every 5 seconds by a participant on average. Under the assumption that a training data set of roughly 5000 observations is required to accurately train the model (allowing 20% for testing) it would take roughly 7 hours to manually label a functional training set.

8.4.4 Market Impact Annotation

Another proposition as to how to produce the training data corpus is through labelling based on the following reaction of the market. In this scenario a headline is provided with its sentiment polarity based off the security price movement following its release, if the price movement is insignificant i.e., ≤ ± 1.5%, then the headline is be annotated with a ‘neutral’ label.

Example: *GOOGL has a fall of -3% thus, any headlines relevant to google from the day prior will be given a ‘negative label.*

The problem with this method is that the sentiment is not directly being evaluated. Difference in opinion by differing publications and writers means that the sentiment polarity of headlines targeted towards a security on one specific day would not be the same, meaning some headlines are always incorrectly annotated. Also, the market is not a direct implication of financial news headlines, there are multiple other factors that affect the price of a security. The combination of all these elements leads to a vast inaccuracy in the annotation of headlines, ultimately causing the classifier to be incorrectly trained.

8.4.5 External Training Data

The selected solution was to use an external annotated dataset for training, this method removes the task of annotation, while still retaining the data quality and integrity required. The dataset used is comprised of ~5000 samples from financial news texts and company press releases that are tagged as positive, negative, or neutral by a group of 16 annotators with adequate business education background. This dataset was made available by Malo, P et al in a 2014 study.

External training data gathered from the following Kaggle dataset:
https://www.kaggle.com/ankurzing/sentiment-analysis-for-financial-news

The dataset is comprised of 4837 unique sentiment annotated financial news headlines, of which 59% are neutral, 28% are positive and 13% are negative. This data set is large enough that its beneficial to the classifier however its balance in sentiment polarity could be improved, however CV training should negate the effects of this. It is important that classes in a dataset are well balanced as imbalanced datasets can lead to models appearing to perform to a high level of accuracy, when in fact this is simply the result of an imbalanced class distribution. This is referred to as the accuracy paradox (Uddin 2019).

8.5 Model Selection and Testing

8.5.1 Methodology

The ‘NLTK’ and ‘scikit-learn’ python modules both provide a variety of built-in ML algorithms which can be used as to classify the sentiment of the headlines. After consideration of the models, three are selected for further evaluation; Naïve Bayes, SVM and Logistic Regression.

Naïve Bayes utilizes the *‘nltk.classify.naivebayes()’* function from NLTK.

To find the probability for a label the algorithm first uses Bayes rule to express **P(label|features)** in terms of **P(label)** and **P(features|label)**:

$$P(\text{ label } \mid \text{ Features }) = \frac{P(\text{ label }) \cdot P(\text{ Features } \mid \text{ label })}{P(\text{Features})}$$

The algorithm then makes the ‘naïve’ assumption that all features are independent, given the label:

$$= \frac{P(\text{ label }) \cdot P(f_1 \mid \text{ label }) \cdot \ldots \cdot P(F_n \mid \text{ label })}{P(\text{ Features })}$$

Rather than computing **P(features)** explicitly, the algorithm just calculates the numerator for each label, and normalizes them so they sum to one.

(Loper 2022)

Support Vector Machine utilizes the *'SVC()'* function from 'skit-learn' applied with the *'nltk.classify.sciklearn()'* wrapper package, allowing a scikit-learn estimator object to be constructed. Support Vector Classification (SVC) is a class variation of SVM capable of performing binary and multi-class classification on a dataset.

SVM functions by taking all data points and defining a 'hyperplane' to divide the classes. This line is termed the decision boundary. Observations can then be classified based off what side off the decision boundary they fall on.

Logistic Regression utilizes the *'LogiticRegression()'* function from 'scikit-learn'.

The hypothesis of logistic regression is calculated with the following formula (returns a value between 0 and 1):

$$h\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

A threshold value is then determined as to how to classify individual observations. Observations are passed through a probability function that allocates a value from 0 to 1. They are then classified depending on what side of the decision boundary they lie.

8.5.2 Evaluation Measures

Implementing the testing procedure is executed through the production of a function that takes the classifier being evaluated and the testing data as parameters. The test data is split into its individual features and labels, the predicted labels are produced using the classifier passed as the argument into the function. Four metrics are evaluated (accuracy, recall, precision and the f_1 score) and displayed as well as a visualisation of the confusion matrix. The measures are calculated using the *'sklearn.metrics'* API to implement functions that are be utilized to assess prediction error.

```
"""Classifier Testing"""

import sklearn
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_score, ConfusionMatrixDisplay

def evaluationMeasures(classifier, testData):
    #Confusion Matrix
    testFeatures = [feature for (feature, label) in testData]
    testLabels = [label for (feature, label) in testData]
    testPredicted = [classifier.classify(feature) for feature in testFeatures]
    confusionMatrix = sklearn.metrics.confusion_matrix(testLabels, testPredicted)
    display = ConfusionMatrixDisplay(confusion_matrix = confusionMatrix)
    #Alternate Metrics
    accuracy = accuracy_score(testPredicted, testLabels) #Accuracy
    recall = recall_score(testPredicted, testLabels, average=None) #Recall
    precision = precision_score(testPredicted, testLabels, average=None) #Precision
    flScore = f1_score(testPredicted, testLabels, average=None) #F1 Score
    #Output
    display.plot()
    print(" ")
    print("Accuracy: " + str(round(accuracy, 2)))
    print("Recall: " + str(round(recall[0], 2)))
    print("Precision: " + str(round(precision[0], 2)))
    print("F1 Score: " + str(round(flScore[0], 2)))
    print(" ")
```

A baseline accuracy of 0.5 is established using a stratified dummy classifier, the more complex classifiers must score higher than this baseline to demonstrate they have any ability to classify sentiment. The dummy classifier makes predictions that ignore the input feature values to fit and predict. The stratified method randomly samples one-hot vectors from a multinominal distribution parametrized by the empirical class prior probabilities.

Metric Results				
Classifier	**Accuracy**	**Recall**	**Precision**	**F_1 Score**
Naïve Bayes	0.84	0.72	0.78	0.75
SVM	0.93	0.97	0.72	0.83
Logistic Regression	0.92	0.93	0.84	0.88

Classifier Confusion Matrices:

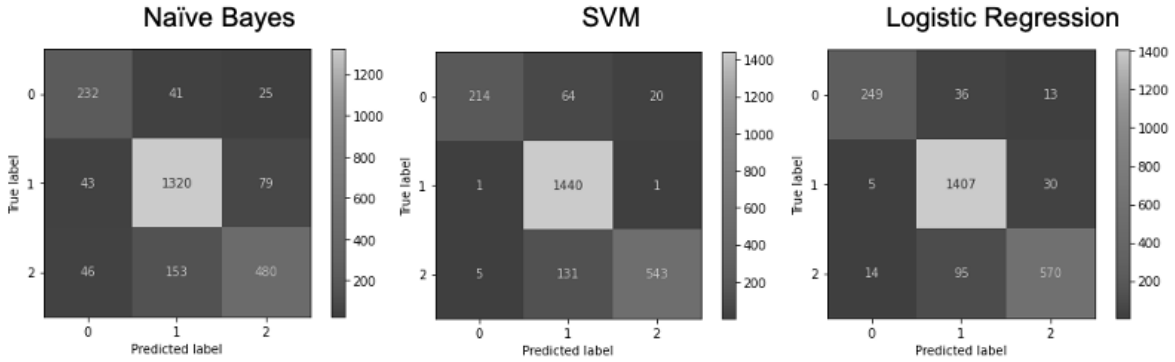

Figure 13: Confusion Matrices; Naïve Bayes, SVM, Logistic Regression

All classifiers clearly beat the established bassline metric (Accuracy > 0.5) and are a viable choice to implement within the final pipeline.

1st Place: SVM
2nd Place: Logistic Regression
3rd Place: Naïve Bayes

Rankings are based off the Accuracy Metric.

SVM is selected to proceed further with through hyper parameter optimisation as it provides more opportunity to further improve (to be described in section 8.6.6). If no metric improvements are made post optimisation, then Logistic Regression can be utilised. Logistic Regression has an accuracy only one percent point lower than SVM but has a superior F_1 Score and a better recall-precision balance.

8.5.3 Cross Validation

A CV function is built to allow the resampling procedure to be used when evaluating each of the ML models under consideration. CV is essential to utilise for this problem as it allows all the training data to be used for training and testing, this supports the evaluation of the algorithm as the data set would be restricted in size if split into a

typical test-train split (i.e., 80-20). Furthermore, additional metrics are gained which support in fine tuning and optimisation of model parameters. The function takes the classifier being evaluated and the training data as parameters and then returns the evaluated classifier. To perform CV the K-Fold process is implemented, the 'sckit-learn' K-Folds cross validator is used to provide train/test indices to split data in train/test sets. The dataset is split into K (K=10) consecutive folds (without shuffling) and the folds are then recursively looped over; each is used once as a validation while the K – 1 remaining folds form the training set remain.

```
"""Cross Validation"""

import sklearn.model_selection

def crossValidation(classifier, trainingData):
    KFoldCV = sklearn.model_selection.KFold(n_splits=10) #10 Folds
    KFoldAccuracy = []
    split = 1
    for trainIndex, testIndex in KFoldCV.split(trainingData): #Fold Testing
        classifier = classifier.train(trainingData[trainIndex[0]:trainIndex[len(trainIndex) - 1]])
        split += 1
    return classifier
```

Executing CV provides a drastic improvement in performance for the ML algorithm due to it providing the ability to train on all the data.

Example: *Under a 50-50 test-train split naïve bayes has an accuracy of 0.71 and F1 score of 0.5. After performing CV, naïve bayes has an accuracy of 0.85 (19.71% increase) and F1 score of 0.76 (52% increase).*

8.6 Model Implementation

8.6.1 Selected Model

Selection: Support Vector Machine

An SVM classifier was the optimal model to implement within the pipeline. SVM scored highest in each individual evaluation metric with a high overall accuracy of 0.93, in further support every metric evaluated scored above 0.7. SVM also allows opportunity to further improve metric results through the tuning of the kernel hyperparameter. The SVM model of classification works well for sentiment analysis and text categorization in general due to its ability to handle a high dimensional input space, which is important when using the BoW model for feature representation. SVM's use overfitting protection, which does not necessarily depend on the number of features, thus they have the potential to handle these large feature spaces. The assumption that most the features are irrelevant in competing classifiers are their pitfall, as aggressive feature selection tends to result in a loss of information. The competing classifiers such as naïve bayes and logistic regression are also 'additive' algorithms, meaning they are better suited for dense concepts rather than sparse document vectors. Finally, the sentiment analysis problem is linearly separable and the theory behind SVM's is to find linear separators (Joachims 1998), deeming it highly suitable.

8.6.2 Training

The model is trained from scratch using CV to utilize all the training data available and produced a drastic improvement in performance in comparison to a basic test-train split. Despite this an 80-20 test-train split is utilized as a benchmark when constructing a stratified dummy classifier, when fitting a scalar to perform standardization and when using learning curves to diagnose ML performance. This text train split is father split into its individual features and labels, designated by X and Y.

```
"""Test-Train Split"""

#80-20 Split
onePercent = len(structuredLabledData)//100
trainingData, testingData = structuredLabledData[onePercent*80:], structuredLabledData[:onePercent*20]
xTrain, yTrain, xTest, yTest = [], [], [], []
for observation in trainingData:
    xTrain.append(observation[0])
    yTrain.append(observation[1])
for observation in testingData:
    xTest.append(observation[0])
    yTest.append(observation[1])
```

Training can be proved to have been successful and had a measurable impact through displaying the most informative features, these are the words that have the greatest impact on the what the classifier determines the polarity to be.

```
                rose = 1.0           positi : neutra =     52.3 : 1.0
                 won = 1.0           positi : neutra =     30.0 : 1.0
                grew = 1.0           positi : neutra =     27.2 : 1.0
         temporarily = 1.0           negati : neutra =     23.2 : 1.0
           temporary = 1.0           negati : neutra =     23.2 : 1.0
                fell = 1.0           negati : neutra =     22.2 : 1.0
                loss = 1.0           negati : neutra =     20.7 : 1.0
               lower = 1.0           negati : neutra =     20.6 : 1.0
                   x = 1.0           negati : neutra =     20.1 : 1.0
                down = 1.0           negati : neutra =     19.4 : 1.0
```

8.6.3 Feature Extraction

To allow the classifier to process the raw natural language within the headlines the text data must be converted into a numerical format based off the individual features/ words it contains. Each document is converted into a vector with one entry for each token in the vocabular that reflects the token's relevance to the document. In turn this constructs a document-term matrix from each individual document.

Implementing this involves the production of a function that takes the pre-processed list of words from a document and the BoW list as parameters. The frequency of individual words within the document are counted and attributed to their corresponding tokens within the BoW representation using a dictionary data structure. The extracted features are then returned from the function.

```
"""Feature Extraction"""

def featureExtraction(words, document):
    wordFrequency = dict.fromkeys(words, 0)
    for word in document:
        if word in words:
            wordFrequency[word] = document.count(word) #Frequency Of Occurrence
    return wordFrequency
```

8.6.4 Feeding and Standardising Data

To enable data to be fed into the classifiers it must be transformed from csv format it is stored in, into a list containing tuples with a dictionary (representing the features in string format, and corresponding value in integer format) and a string (representing the label). This is achieved through a nested loop to extract the individual data values.

```
"""Feeding"""

labledData = pandas.read_csv('Data/Training/all_labled_data.csv')

#Classifier Training Format
structuredLabledData = []
for index, row in labledData.iterrows():
    features = {}
    sentiment = row['Sentiment']
    for feature in wordSet:
        features[feature] = row[feature]
    structuredLabledData.append((features, sentiment))
```

Standardising the data gives all the features the same influence on the distance metric within an SVM, preventing large features dominating over the others and skewing the results. To standardize the training data the '*StandardScalar*' function from '*sklearn.preprocessing*'. The function works by standardizing features by removing the mean and scaling to the unit variance. Centring and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set.

```
"""Standardisation"""

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

numericFeatureList = []
for values in xTrain:
    numericFeatures = list(xTrain[0].values())
    numericFeatureList.append(numericFeatures)

#Standardize Data
scaler.fit(numericFeatureList)
```

Mean and standard deviation are used within the '*transform*' function, which performs the process.

8.6.5 Model Diagnostics

To analyse the SVM algorithm, three diagnostic visualisations are produced: Performance Learning Curve, Model Scalability Graph, and Model Performance Graph. Gathering the necessary data to generate these visualisations was conducted through the '*sklearn.model_selection.learning_curve*' function. It operates through determining cross-validated training and test scores for different training set sizes for a given dataset and ML model.

A cross-validation generator splits the whole dataset k (k=10) times in training and test data. Subsets of the training set with varying sizes are used to train the estimator and a score for each training subset size and the test set are computed. Afterwards, the scores are averaged over all k runs for each training subset size. The training sizes, training scores, testing scores, fitting times and score times are then returned.

The learning curve is produced through plotting training scores (Train) and testing scores (Validation) against the training sizes. The shape and dynamics of a learning curve are used to diagnose the behaviour of a ML model and in turn suggest the type of configuration changes that are made to improve the performance.

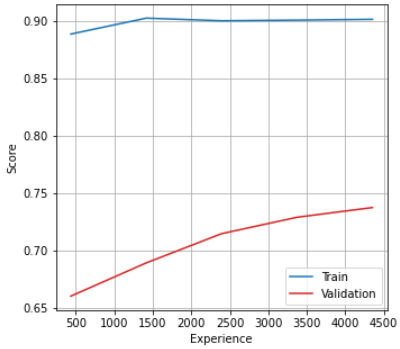

Figure 14: Support Vector Machine learning curve.

Both the train and the validation lines can be seen to be begging to converge together however would benefit from more training data to generalize more efficiently or alternately a model that accounts for the data being linearly separable.

The model scalability graph is produced through plotting the fitting time against the training sizes. This demonstrates the times required by the model to train various sizes of training dataset.

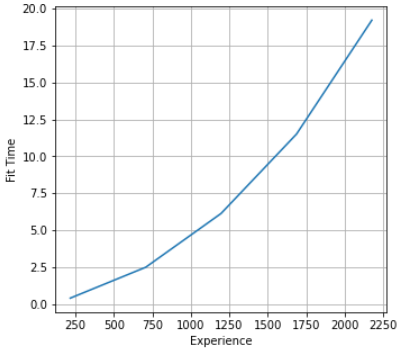

Figure 15: Support Vector Machine scalability graph.

The model scalability curve is clearly demonstrating exponential growth, as the model trains on more observations the time required to fit the model increases without bound. This would cause issue when scaled to a larger training data set, however for this given problem scale it remains viable.

The model performance graph is produced through plotting the testing scores against the fitting time. This demonstrates how much time is required to train the model for each training size.

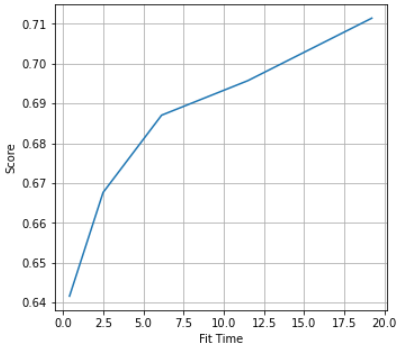

Figure 16: Support Vector Machine performance graph.

The model performance curve can be seen to be converging to a set performance as the time taken to fit increases. This is expected and when fitted on the entire training data set through cross-validation causes no issues, residing at an accuracy of 0.93.

8.6.6 Model Optimisation

To improve the performance of a ML model the hyperparameters can be tuned, SVM's core hyperparameter is the 'Kernel', the kernel is responsible for mapping the observations into some feature space, ideally making the observations more separable after this transformation

As default 'Sciklearn' uses the Radial Bias Function (RBF) Kernel for its SVC.

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

The RBF Kernel is a non-linear Kernel however this problem uses linearly sparable data so the Linear Kernel should be the optimal choice.

$$k(x, y) = 1 + xy + xy\min(x, y) - \frac{x + y}{2}\min(x, y)^2 + \frac{1}{3}\min(x, y)^3$$

A 'Linear' Kernel can be used a solution through the implementation of the 'LinearSVC' (Linear Support Vector Classification) function from 'skit-learn' applied with the *'nltk.classify.sciklearn()'* wrapper package. This provides more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples. A 'Linear' kernel is well suited for text classification as it thrives when dealing with large sparse data vectors and takes less training time compared to other kernel functions. This adjustment resolves the issues identified through the diagnostic visualisations produced such as the slow convergence of the training and validation lines as well as the steep exponential scalability curve. This is the best suited Kernel parameter for this problem.

Linear SVM Metric Results			
Accuracy	**Recall**	**Precision**	**F_1 Score**
0.97	0.97	0.94	0.95

A significant improvement in all evaluation metrics is recorded, with all values recorded being higher than 0.9. A significant 4.3% increase in accuracy can be seen as well as a 14.5% increase in F_1 score. This classification model with this hyperparameter setting will be used in the final pipeline.

```
"""Optimising Model (Hyper Parameter Tuning - Kernel)"""

from sklearn.svm import LinearSVC

svmClassifierOptamised = nltk.classify.SklearnClassifier(LinearSVC())
svmClassifierOptamised = crossValidation(svmClassifierOptamised, structuredLabledData)
print("Support Vector Machine Classifier: ")
evaluationMeasures(svmClassifierOptamised, testingData)
```

8.7 Full Pipeline

Combining various sub functions built throughout the code a pipeline is produced into a function through a four-stage process, taking a singular document (headline) and returning it back along with the securities it references and the sentiment polarity it conveys.

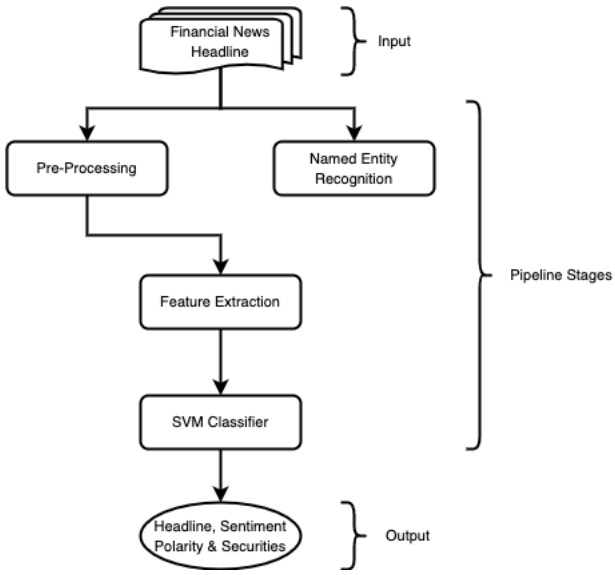

Figure 17: The classification pipeline.

When a financial news headline is passed into the function it is initially stored as a local variable within the function so its original instance can be used at multiple stages. The original instance will undergo NER to identify securities pretraining to the headline, so they can be returned by the function, however the original instance will not be altered. A new instance of the headline is created and undergoes pre-processing and then feature extraction, to make it readable by the ML classifier for the final stage where it is passed into the trained SVM classifier and assigned a polarity. The original headline, sentiment polarity and identified securities are then returned.

To execute this pipeline for all 53118 headlines the combined dataset of financial news headlines is iterated over, using a ‘for loop’. Each headline is processed and the headline, sentiment, securities, and corresponding date the headline was published are written into a new data frame that is stored as a CSV file.

```
"""Sentiment Processing"""

import time

#Data Acsess
unclassifiedData = pandas.read_csv('Data/all_headlines.csv')
unclassifiedData = cleanData(unclassifiedData)

#Data Frame Construction
classifiedData = pandas.DataFrame(data={'Document': [], 'Securities': [], 'Sentiment': [], 'Date': []})

headlines = unclassifiedData["Headlines"].to_list()

start = time.time()
for row in range(0, len(headlines)): #Headline Processing
    entry = classificationPipeline(headlines[row])
    entry.append(pandas.to_datetime(unclassifiedData["Date"][row]))
    classifiedData.loc[len(classifiedData)] = entry
    classifiedData.to_csv('Data/classified_data.csv')
end = time.time()

print("Runtime: " + str(round(end - start) 2))
```

Processing each headline takes around 5000 seconds to complete, on typical hardware. Once sentiment processing is completed EDA is conducted on the results to investigate whether they are viable to perform statistical analysis on, the distribution of polarity is measured to gain insight into if a large enough number of subjective headlines exist to test the hypothesis. The objective headlines are filtered out as they are of no use in the statistical analysis.

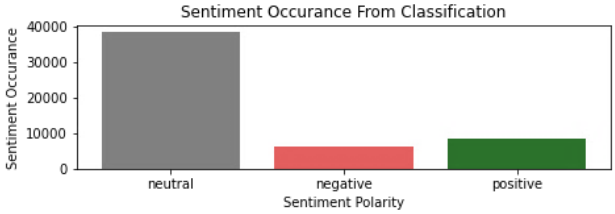

Figure 18: Bar Chart detailing the classified sentiment distribution.

A total of 14652 subjective headlines were classified with the majority being negative. This is enough headlines to conduct a correlation analysis on the overall US market however it may be insufficient for headlines relating to individual securities.

Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
“Ackman exits Nike stake with $100 million in profit: source”	Positive	Positive
“Google defeats lawsuit claiming YouTube censor’s conservatives”	Neutral	Positive
“Saudi Arabia’s new crown prince to sit down with wall street”	Negative	Neutral
“GameStop tops revenue estimates on robust Nintendo Switch sales”	Positive	Positive

“U.S. opens probe into fatal Tesla crash, fire in California”	Negative	Negative

Through basic human analysis of the classified headlines, most headlines that contained subjective sentiment were falsely classified as neutral where there was clear negative or positive polarity present.

8.8 Statistical Analysis

8.8.1 Market Overview

Two core values are required for analysis: the average sentiment and the normalized market price. To calculate these values two functions are produced ‘calculateAverageSentiment’ and ‘normalizeMarketData’.

calculateAverageSentiment(sentiments) - The average sentiment calculation is achieved by assigning a value of 1 for every positive sentiment given in the parameter list passed into the function, the sum is then calculated and divided by the number of sentiments passed in.

```
def calculateAverageSentiment(sentiments):
    totalSum = 0
    for sentiment in sentiments:
        if sentiment == 'positive': #Value Assignment
            totalSum += 1
    average = round(totalSum/len(sentiments), 3) #Average Calculation
    return average
```

normalizedMarketData(marketData) - The normalized market data calculation is achieved by taking a list of market data values and using the minimum value and maximum value to proportionally change each value to its equivalent between 0 and 1.

```
def normalizeMarketData(marketData):
    marketData = (marketData - numpy.min(marketData)) / (numpy.max(marketData) - numpy.min(marketData)) #Normalize
    normalizedPriceList = []
    for price in marketData: #Cleaning List
        normalizedPriceList.append(price)
    return normalizedPriceList
```

To analyse securities two separate visualisations are produced to gain a perspective on how or if the sentiment of financial news headlines directs and reflects the performance of the US stock market. The individual visualizations are built into individual sub functions and an encompassing function produces the graphics as well as return a correlation estimate. This larger function takes a ticker symbol as a parameter so it can be utilized for multiple securities.

Monthly Sentiment and Monthly Security Price Time Series:

The production of the time series will be an attempt to analyse how the prevailing sentiment directs the market. A monthly time interval is used between data points to allow for more sentiments to be processed hence improved accuracy, to account for this the market close price every 21 trading days will be taken as the comparison, the values are normalized using the ‘normalizeMarketData’ function. The monthly average sentiment is calculated through taking the sentiment of the corresponding

financial news headlines for that month and passing it into the ‘calculateAverageSentiment’ function. Due to the insights of the EDA into distribution the first two months (December 2017 & January 2018) are removed from the time series scope as there is not sufficient data to produce an accurate portrayal of the sentiment in that period.

Daily Sentiment Value against Daily Security Price Change Scatter Plot:

The production of the scatter plot will be an attempt to analyse how the prevailing sentiment directs the market. A daily time interval is used is used between data points to improve the accuracy of the linear regression line and thus the correlation. The market price change from the price change from the previous day is calculated and is shifted to correspond to the sentiment from the day prior, the price change is then normalized using the ‘normalizeMarketData’ function. The daily average sentiment is calculated through taking the sentiment of the corresponding financial news headlines for that day and passing it into the ‘calculateAverageSentiment’ function.

The encompassing function takes the values produced by each sub function to plot two the two figures. On the ‘Daily Sentiment Value against Daily Security Price Change Scatter Plot’ a linear regression line is drawn using the ‘linregress’ function.

8.8.2 Individual Securities

As fewer headlines reference individual securities than the overall market there is significantly less data to analyse for them. A threshold of 50 subjective headlines is decided, to have a moderate level of accuracy when conducting statistical analysis.

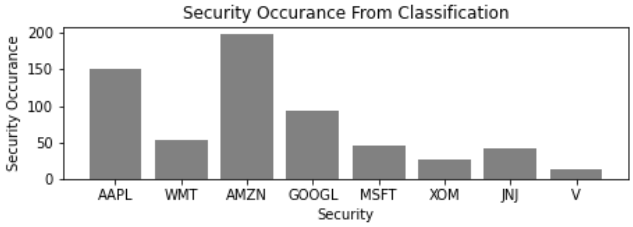

Figure 19: Bar Chart detailing the security frequency distribution.

As portrayed in the figure detailed, only Apple (APPL), Walmart (WMT), Amazon (AMZN), Google (GOOGL), and Microsoft (MSFT) have sufficient headlines to semi-accurately calculate any potential correlation.

8.8.3 Correlation

As the hypothesis references the financial news headline sentiment influencing both the direction and reflection of the U.S. stock market, each aspect requires testing through a consistent measure. The correlation coefficient for each is calculated through the gradient of a linear regression line. This calculation works as correlation quantifies the strength of a linear relationship between a pair of variables. Thus, the gradient of a linear regression line directly demonstrates the strength of the variable’s relationship and therefore its correlation. The linear regression line is

calculated for both visualisations however it is only visually represented through the results on the direction scatter plot.

The gradient and linear regression line values are produced through the '*scipy.stats.linregress'* function, which calculates a linear least-squares regression for two sets of measurements.

8.8.4 Significance Test

To conduct the significance test and calculate the T-Value and P-Value for the data points under consideration the '*scipy.stats.ttest_ind*' function is implemented. The independent samples of scores are taken as parameters and a two-sided test for the hypothesis is conducted returning a numeric T-Value and P-Value. The P-Value is then used to either accept the null or alternate hypothesis.

Due to the nature of the data under consideration a disproportionate amount of project time went towards researching and implement an accurate method of statistical analysis, through a significance test, to evaluate the hypothesis. However, this was resolved through saving time in sections that were easier to implement than initially perceived.

9 Results and Evaluation

9.1 Results

Two significance tests are conducted to evaluate the following hypotheses:

Direction H_0 Null Hypothesis: The sentiment of financial news headlines does not direct the performance of a U.S. security.

Reflection H_0 Null Hypothesis: The sentiment of financial news headlines does not reflect the performance of a U.S. security.

Significance Level: $\alpha = 0.05$ (5%)

9.1.1 S&P 500

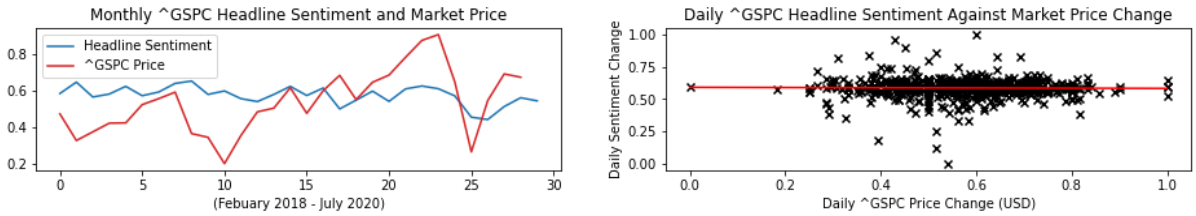

Figure 20: S&P 500 hypothesis evaluation results.

A clear reflection is displayed in the time series between the headline sentiment and the price, with the movement of headline sentiment mitigated in size in comparison, despite this the pattern directions still demonstrate similar trends throughout. There is clearly no positive correlation relationship between the sentiment and change in security price.

S&P 500 Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'U.S. trade deficit at three-year low as imports tumble'	Neutral	Negative
'Ford's China vehicle sales drop 26% in third straight year of decline'	Negative	Negative
'U.S. consumer prices gain slightly; underlying inflation tame'	Positive	Positive

<u>Market Direction Hypothesis Test:</u>

Correlation Coefficient: -0.01
T-Value: -2.00
P-Value: 0.05

The P-Value meets the significance level ($\alpha \leq 0.05$), however there is only negligible correlation present.

Accept Direction H_0 Null Hypothesis

Market Reflection Hypothesis Test:

Correlation Coefficient: 0.27
T-Value: 1.27
P-Value: 0.21

The P-Value meets the significance level ($\alpha \leq 0.21$) and there is a clear correlation present.

Reject Reflection H_0 Null Hypothesis

Accept Reflection H_a Alternate Hypothesis (The sentiment of financial news headlines does reflect the performance of a U.S. security).

9.1.2 Apple

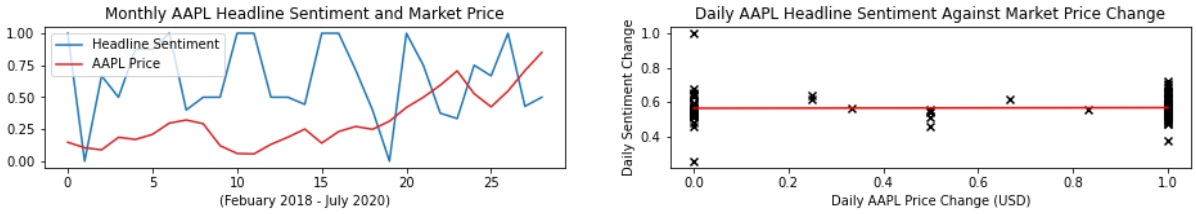

Figure 21: Apple hypothesis evaluation results.

No reflection can be observed between the headline sentiment and price of the security. There is clearly no positive correlation relationship between the sentiment and change in security price. A lack of data was recorded for this security thus the results may be inaccurate.

Apple Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'Jim Cramer says Apple and General Electric 'embraced radical transparency''	Neutral	Neutral
'FTC demands data on small buys by Google, Amazon, Apple, Facebook, Microsoft'	Negative	Neutral
'Apple's iPhone back to growth as company braces for coronavirus impact'	Positive	Positive

Apple Direction Hypothesis Test:

Correlation Coefficient: 0.00
T-Value: 0.76
P-Value: 0.45

The P-Value meets the significance level ($\alpha \leq 0.45$), however there is no correlation present.

Accept Direction H_0 Null Hypothesis

Apple Reflection Hypothesis Test:

Correlation Coefficient: -0.11
T-Value: 5.26
P-Value: 0.00

The P-Value did not meet the specified significance level (α ≥ 0.00).

Accept Reflection H_0 Null Hypothesis

9.1.3 Walmart

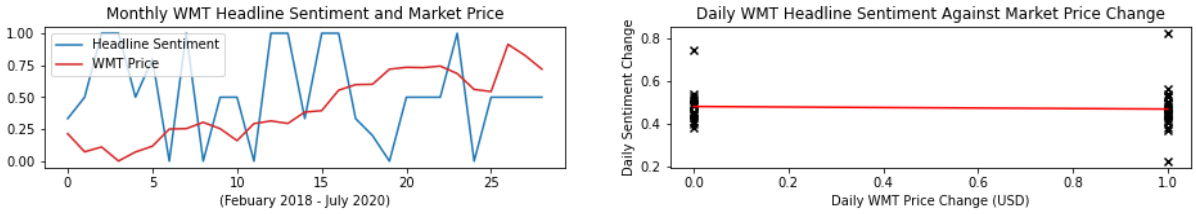

Figure 22: Walmart hypothesis evaluation results.

No reflection can be observed between the headline sentiment and price of the security. There is clearly no positive correlation relationship between the sentiment and change in security price. A lack of data was recorded for this security thus the results may be inaccurate.

Walmart Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'Walmart sees slower online sales growth after tepid holiday quarter'	Neutral	Negative
'Walmart opens first small high-tech supermarket in China'	Negative	Neutral
'Walmart uses massive Trump tax gain to offer modest pay rise to workers'	Positive	Positive

Walmart Direction Hypothesis Test:

Correlation Coefficient: -0.01
T-Value: 1.08
P-Value: 0.29

The P-Value meets the significance level (α ≤ 0.29), however there is only negligible correlation present.

Accept Direction H_0 Null Hypothesis

Walmart Reflection Hypothesis Test:

Correlation Coefficient: -0.15

T-Value: 1.40
P-Value: 0.17

The P-Value meets the significance level ($\alpha \leq 0.17$), however there is only negative correlation relationship present.

Accept Reflection H_0 Null Hypothesis

9.1.4 Amazon

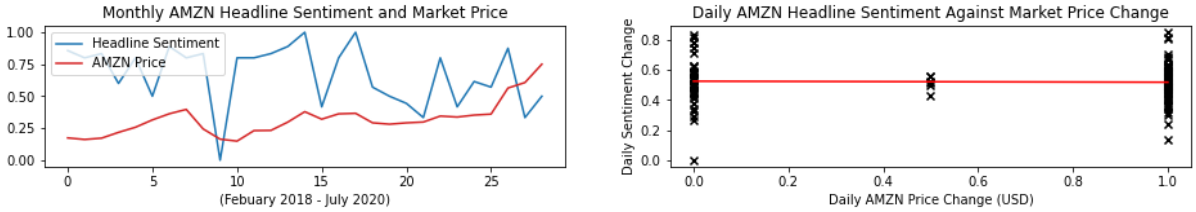

Figure 23: Amazon hypothesis evaluation results.

No reflection can be observed between the headline sentiment and price of the security. There is clearly no positive correlation relationship between the sentiment and change in security price. A lack of data was recorded for this security thus the results may be inaccurate.

Amazon Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'Kshama Sawant wants to tax Amazon so people can afford to live in Seattle. Can she pull it off?'	Neutral	Neutral
'Amazon struggles to halt tide of coronavirus profiteers'	Negative	Negative
'Amazon confirms two employees in Italy have contracted coronavirus'	Positive	Neutral

<u>Amazon Direction Hypothesis Test:</u>

Correlation Coefficient: -0.01
T-Value: 3.91
P-Value: 0.00

The P-Value did not meet the specified significance level ($\alpha \geq 0.00$).

Accept Direction H_0 Null Hypothesis

<u>Amazon Reflection Hypothesis Test:</u>

Correlation Coefficient: -0.00
T-Value: 7.50
P-Value: 0.00

The P-Value did not meet the specified significance level ($\alpha \geq 0.00$).

Accept Reflection H_0 Null Hypothesis

9.1.5 Google

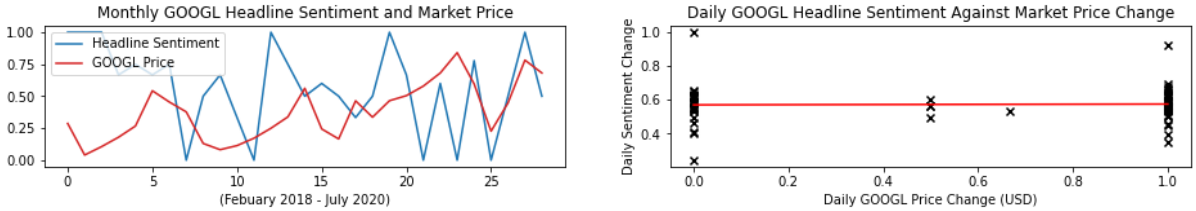

Figure 24: Google hypothesis evaluation results.

No reflection can be observed between the headline sentiment and price of the security. There is clearly no positive correlation relationship between the sentiment and change in security price. A lack of data was recorded for this security thus the results may be inaccurate.

Google Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'Google cancels annual developer conference on coronavirus fears'	Neutral	Neutral
'Facebook, Google ask San Francisco staff to work from home as coronavirus spreads'	Negative	Neutral
'Google, Walmart join U.S. effort to speed up coronavirus testing'	Positive	Neutral

Google Direction Hypothesis Test:

Correlation Coefficient: 0.00
T-Value: 1.12
P-Value: 0.27

The P-Value meets the significance level ($\alpha \leq 0.27$), however there is no correlation present.

Accept Direction H_0 Null Hypothesis

Google Reflection Hypothesis Test:

Correlation Coefficient: -0.08
T-Value: 2.75
P-Value: 0.01

The P-Value did not meet the specified significance level ($\alpha \geq 0.01$).

Accept Reflection H_0 Null Hypothesis

9.1.6 Microsoft

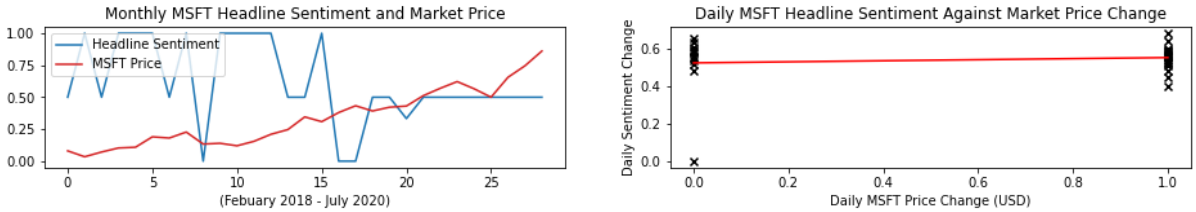

Figure 25: Microsoft hypothesis evaluation results.

No reflection can be observed between the headline sentiment and price of the security. There is clearly no positive correlation relationship between the sentiment and change in security price. A lack of data was recorded for this security thus the results may be inaccurate.

Microsoft Classified Headlines		
Financial News Headlines:	Classified Sentiment	Human Sentiment
'Remote work revenue could help Microsoft offset coronavirus impacts; analysts say'	Neutral	Positive
'Microsoft to invest $1 billion in Polish cloud project'	Negative	Neutral
'Microsoft 'might be the best tech stock in this market,' Jim Cramer says'	Positive	Positive

Microsoft Direction Hypothesis Test:

Correlation Coefficient: 0.03
T-Value: 1.24
P-Value: 0.22

The P-Value meets the significance level ($\alpha \leq 0.22$), however there is only negligible correlation present.

Accept Direction H_0 Null Hypothesis

Microsoft Reflection Hypothesis Test:

Correlation Coefficient: -0.28
T-Value: 4.15
P-Value: 0.0

The P-Value did not meet the specified significance level ($\alpha \geq 0.00$).

Accept Reflection H_0 Null Hypothesis

9.1.7 Result Conclusion

The study was able to demonstrate the hypothesis to an extent, showing that the sentiment of financial news headlines relating to the overall U.S. market directly reflects the price (USD) of the S&P 500, monthly. This was proved through the acceptance of the alternate reflection hypothesis for the S&P500. Despite this no correlation between the price of an individual stock and the sentiment of directly

relating financial news headlines could be found, as the null hypothesis was accepted. Additionally, there was no evidence to suggest that the daily sentiment of a security had any influence over its corresponding price change the subsequent day, for any security, with the inclusion of the broader market. No individual stocks rejected either null hypothesis for both reflection and direction as either the associated P-Value did not meet the significance level or there was only negligible correlation present.

9.2 Evaluation

To evaluate the system in its entirety each core element has been discussed with reference to its corresponding functional requirement if present. The elements have been evaluated against a derived performance metric, through various rigorous testing procedures as described, to determine their success. This section aims to discuss the strengths and weaknesses of the project and any adjustments / improvements that would be made if given a limitless time frame. Moreover, a critical appraisal of the whole project is provided to indicate the confidence level about what has been produced.

9.2.1 Data Collection

The three datasets under examination to be classified ware required to contain enough subjective financial news headlines targeted towards securities, that had a clear sentiment polarity present, so that an accurate numeric representation of the sentiment towards an individual security on an individual day can be calculated. A clear performance measure to evaluate the quality of headlines of use is to check the number of subjective headlines identified relevant to securities. 14652 subjective headlines were identified by, as all headlines are considered relevant to the (27.6% of the total dataset). While this is still enough data to provide insight into the sentiment surrounding a security, it would be beneficial to scale this study to a larger dataset to improve the accuracy of the findings.

External training data used to train the ML algorithm was required to be of large enough quantity that the model can be thoroughly trained and of higher enough quality so that it can be accurately trained. Training data quality considerations involve the accuracy of the annotation as well as the sentiment class balance within the dataset. The advantages of this external data set are its direct relevance to the data being evaluated, as both are specifically ‘financial’ news texts, but also its quality, as a group of 16 annotators with adequate business education background were responsible for producing it. In contradiction, the dataset is too small to fully train a ML algorithm and is unbalanced (59% are neutral, 28% are positive and 13% are negative) leading to a potential accuracy paradox. To improve upon the external dataset, the training data could be combined with manual annotation conducted upon the dataset under evaluation to balance out the classes and build upon its size.

The core purpose of the ‘yFinance’ API was to gather the accurate daily close prices (USD) of a specified security from the 22nd of December 2017 to the 18th of July 2020. Despite this working as a solution short term, downloading market data can be pragmatic through the Yahoo Finance API and sometimes throws an error. This is due to Yahoo’s recent acquisition and the old API (used within the project), slowly

being phased out, and being replaced with a new system. To resolve this issue an alternate finance API can be implemented as a long-term solution.

9.2.2 Data Processing

The objective of cleaning the dataset was to fix any data that is incorrect, inaccurate, incomplete, incorrectly formatted, duplicated or irrelevant to conducting the sentiment analysis of financial news headlines. A clear performance measure for the cleaning process is the size reduction of the dataset. Initially the combination of the three data sets contains 53158 entries, however post cleaning the dataset is reduced to 53118 entries, mainly due to datetime null values (without which, no information can be derived from the data). This demonstrates that the cleaning process is clearly functional, and with human evaluation using the SCP tool it can be clearly seen that the data is of superior quality (uniform format and structure).

Pre-Processing was conducted when producing the BoW model and prior to feature extraction to reduce the dimensionality of the data being fed into the model and to abstract any irrelevant data from the model. This corresponds directly to the functional requirement that the project MUST utilize basic pre-processing techniques. To evaluate the achievement of this the process the dimensionality reduction can be calculated as a performance metric. Prior to pre-processing the size of the BoW model was 23068 words after it is 10235 words, a reduction of 55.63%, losing information with only a negligible impact. Unfortunately, this process is inefficient, taking roughly 3000 seconds to complete (hardware dependent). The impact of this was mitigated through storing files externally to the development environment.

NER was conducted to identify securities within the headlines so they could be removed, to prevent overfitting, but still linked to the headline separately when performing analysis at later stages. NER additionally aimed to replace all named entities within the headlines with their corresponding categoric title. This corresponds directly to the functional requirement that the project MUST implement NER Recognition for individual securities. This can be evaluated using the SCP to search for individual securities and entities within the processed text. Post NER, upon generating a word list (with the SCP) containing the names of all securities, and any alternate names, when generating concordances and searching no results can be found. Additionally, when manually reading headlines at random very few entities can be detected, only the few that were missed by the Spacy NER library. Overall, the NER two-step function was highly effective at removing entities and fully effective at removing those relevant to the problem at hand.

Standardising the data gives all the features the same influence on the distance metric within an SVM, preventing large features dominating over the others and skewing the results. Unfortunately, when testing this procedure had no impact on the evaluation metrics so was not used in the final system.

9.2.3 Data Analysis

Evaluation Measures were used to give insight into the performance of the ML model. This corresponds directly to the functional requirement that the project

SHOULD test for an optimal ML algorithm. The evaluation measure's function is used at various stages of the study to provide an accurate insight into the performance of the classifier provided from the data passed as a parameter. To improve upon the evaluation measure's function, additional metrics could be considered, such as Logarithmic Loss, Mean Absolute Error, and Mean Squared Error, this would provide a better picture of the model's fulfilment of the classification problem.

CV was utilised to enhance the training process, through allowing the model to train on the entirety of the training data set, as well as to produce diagnostics to evaluate the ML model. The cross-validation process can be evaluated by the margin at which the classifier evaluation measures have improved after implementing the process. In practice, implementing CV, compared to typical fitting with a test-train split, improved every metric, proving the resampling method is beneficial to the study. This additionally proves that K-Fold was the optimal choice, rather than Monte Carlo, as training on the entirety of the dataset is what was responsible for the metric improvement.

Diagnostics were produced to expose potential issues with the model and allow the necessary adjustments to be made. This additionally corresponds directly to the functional requirement that the project SHOULD test for an optimal ML algorithm. These diagnostic visualisations were useful to provide an insight into the learning, scalability, and performance of the model. However, no action was required to be taken to adjust the data due to the model being well fitted. Despite this the diagnostics did support the decision making into the kernel parameter decision for the SVM classifier.

Optimisation of the ML algorithm was conducted through the tuning of hyperparameters to improve the performance of the model for the problem at hand. This corresponds directly the functional requirement that the project COULD optimise a ML model through hyper parameter tuning. The hyper parameter tuning can be evaluated by the margin at which the classifier evaluation measures have improved after implementing the process. Implementing the 'Linear' Kernel to the SVM clearly made an impact as there was a significant improvement in all evaluation metrics recorded, a 4.3% increase in accuracy was recorded as well as a 14.5% increase in the F_1 score, thus proving the value of the hyperparameter tuning.

The ML pipeline aimed to provide an end-end construct that orchestrates the into and out of the ML model. This corresponds directly to the functional requirement that the project MUST produce a sentiment classification tool. The pipeline can be tested as to its accomplishment via the results it produces, and the time required to produce such results, this can provide insight into its performance and scalability. From the results produced by the pipeline it is obvious that it functions as expected however it is an inefficient process and would struggle to scale up to a larger dataset, taking roughly 5000 seconds to complete. The impact of this was mitigated through storing files externally to the development environment.

9.2.4 Data Visualisation

A time series produced for an individual security aimed to visualise any reflection between the sentiment surrounding security and its price. This corresponds directly to the functional requirement that the project MUST plot the price and sentiment of individual securities on a time series. The unforeseen issue with this aspect of analysis was the lack of data for the initial two months (discovered from the distribution EDA) thus as there was not sufficient data to produce an accurate portrayal of the sentiment in that period, they were removed from the time series scope. The resultant time series is an accurate representation of the sentiment portrayed towards an individual security and the securities price, monthly.

A scatter plot produced for an individual security aimed to visualise any direction between the sentiment surrounding security and its price. This corresponds directly to the functional requirement that the project SHOULD plot the scatter graph for the price against sentiment with a linear regression line. Unfortunately, not enough subjective security data was gathered from classification to evaluate all the securities under investigation. Thus, a threshold was implemented with a minimum requirement as to how many subjective sentiment values were required for an individual security to accurately evaluate it. Despite this there is still not enough data to accurately produce a plot for most securities, except for the S&P 500, as can be seen through spaced data points on the figures produced (due to insufficient data to produce an accurate average representation). To improve upon this, a larger data set could be considered, or alternatively a dataset with security specific headlines.

The correlation was calculated to provide an accurate numeric representation of the results that could be tested to evaluate the hypothesis. This corresponds directly to the functional requirement that the project MUST calculate the correlation coefficient for the price against sentiment. This was achieved through calculating the gradient of the linear regression line on the scatter plot. This provides an accurate numeric value from which clear testable criterion were produced to evaluate the hypothesis of the study.

9.2.5 Critical Appraisal

To summarise, the project was a success, the approach and implementation to proving the hypothesis were executed to a high standard and could only be further improved in future work. The classifier produced within the study thrived displaying highly impressive evaluation metrics, post optimisation. Successful pre-processing techniques were largely responsible for the achievement of the project. The main attribute that negated the accuracy of the results and caused unexpected issues with the project were the datasets used and means of data collection, due to either their size, quality, or reliability. Alternate limitations do exist, mainly residing in the IR methods employed by the system and the general scalability of the model.

10 Future Work

Despite the achievement of this study, there is always room to further improve what has been implemented and follow on from this research by a third party. The adage of Hofstadter's law that 'Everything takes longer than you think, even when you consider Hofstadter's law' (Law 2009) reigns true in this project. As outlined in the project approach, all 'MUST' and 'SHOULD' functional requirements have been met by the system produced. This section outlines a starting point for the continuation of this line of work.

One 'COULD' functional requirement that was not met by the system is that the ML classifier COULD account for Amplifiers, De-Amplifiers, Negators and Adversative Conjunction. This functional requirement did not occur within this project as the BoW model of feature extraction abstracts meaning, context and order (semantics) from the documents. Context and meaning can offer a lot to the model, that if modelled could tell the difference between the same words differently arranged. To preserve semantics an alternate information retrieval (IR) method must be implemented. Two methods that would be well adapted to this projects scope are 'Word Embedding' and 'GloVe'. Word Embedding represents a document in the form of a real-valued vector that encodes the meaning of a word such that the words that are closer in vector space are expected to be similar in meaning. Global Vector for word representation (GloVe) is an unsupervised learning model for distributed word representation, obtaining vector representation for words through mapping into meaningful space where distance between words represents semantic similarity (Pennington 2014). Both these feature representation avenues would apply the functional requirement and be an adequate next step to improve the performance of the system built.

Another improvement for future development, as aforementioned in the project's critical appraisal (see section 9.2.5), are the datasets used and means of data collection. The datasets under evaluation did not contain an adequate amount of data to provide insight into the sentiment surrounding a security and it would be beneficial to scale this study to a larger dataset to improve the accuracy of the findings. The external training data is too small to fully train a ML algorithm and is unbalanced, leading to a potential accuracy paradox. To improve upon this the external dataset and the training data could be combined with manual annotation conducted upon the dataset under evaluation to balance out the classes and build upon its size. This was also stated as a 'COULD' functional requirement at the beginning of the project, stating that the project COULD conduct a survey to determine the sentiment value of specific terms for labelling.

Scalability will become an issue as datasets under consideration increase in size. This was demonstrated within the scalability learning curve, the model scalability curve is clearly demonstrating exponential growth, as the model trains on more observations the time required to fit the model increased without bound. This would cause issue when scaled to a larger training data set, however for this given problem scale it remains viable. As issues with processing inefficiency have already appeared through timing processes such as the pipeline, it would be required to develop these areas further to account for an increase in data. Where necessary the algorithmic complexity will need to be reduced as well as optimising the ML model to reduce

processing time. The improved system should be feasible to utilize on a dataset roughly ten times the size of the current, while still using basic hardware requirements.

To maximally optimise the system, a deep convolutional neural network could be developed to perform sentiment analysis, replacing the SVM classifier within the pipeline. This would be paired ‘Word Embeddings’ feature representation previously mentioned, or alternately the ‘GloVe’ unsupervised learning strategy could be utilised. Theoretically, this should improve classification accuracy but will require substantial amounts of training data as there is no threshold for learning as with a traditional ML algorithm. An additional suggestion, to improve the classifier further the ‘Gamma’ hyper parameter could also undergo tuning.

Finally, the findings from this research may be implemented in various financial strategies to gain an insight into the drivers behind security price fluctuations, leaving the scope of this project. The sentiment classification tool produced within this project could be reconfigured to provide signals within an algorithmic or standard trading strategy. This would function by scraping financial news headlines in real time and feeding them into the pipeline. Depending on how the headline was classified, and the security it was linked to, it would either add or negate from an overall sentiment value for that security, the headlines impact on the value would decrease over time dependent on how long ago the headlines were published. A significant change in this value may indicate a change in opinion towards a security, prior to the market shifting. They could allow a trader to profit through buying or selling the security in question.

To summarise, there are multiple directions this line of work can take, even branching out from financial news headlines into general financial discussion and forms of dialog. As NLP and ML techniques evolve and adapt this project can move with them, to improve its performance and results.

11 Conclusion

The study was able to demonstrate the hypothesis to an extent, showing that the sentiment of financial news headlines relating to the combined U.S. market directly reflects the price (USD) of the S&P 500, monthly. Despite this no correlation between the price of an individual stock and the sentiment of directly relating financial news headlines could be found. Additionally, there was no evidence to suggest that the daily sentiment of a security had any influence over its corresponding price change the subsequent day. The results of this study would not be sufficient to build towards disproving EMH or 'Random Walk Theory'.

In contradiction to the results, if a larger dataset of financial news headlines was implemented there may be a drastic change in the visualisations and correlation values, particularly for individual securities where there was a lack of data examined.

Project Aims:

- **A1:** Complete research into the various aspects of the project.
- **A2:** Construct an accurate and complete training data set suitable for algorithmic supervision.
- **A3:** Produce a functioning sentiment analysis classification model through machine learning.
- **A4:** Conduct statistical analysis on a minimum of 5 security's including the overarching market.
- **A5:** Complete Documentation and Report.

A1 was achieved at the beginning of the study, through conducting online research, building background with academic journals, online tutorials, and studies, in the first two weeks of the project. **A2** was completed through an external dataset and pre-processing and feature extraction/ representation techniques were executed to train the model. **A3** is produced through a ML supervised classifier, in the form of a SVM with a linear kernel, was implemented to meet this aim. **A4** was conducted through analysis performed on the following stocks: AMZN, WMT, GOOGL, APPL, MSFT, as well as the S&P 500 tracking index. **A5** was achieved through fully documenting the code and writing the report detailing the project process.

The classifier produced within the study thrived displaying highly impressive evaluation metrics, post optimisation. The final classifier having only a 3% error rate, rivalling the human classification error rate (reported to be 5.1%) (Dodge 2017). This was largely due to hyper parameter tuning, the linear kernel of the SVM was well suited for sparse vectors and counteracted the shortcomings of the BoW model. CV prospered within this problem, dramatically improving performance due to the additional data for training it provided.

Successful pre-processing techniques were largely responsible for the achievement of the project. NER supported the fitting of model, as well as the identification of securities for later analysis. The reduction of dimensionality through pre-processing also aided the model's performance significantly, mitigating the curse of dimensionality (Zhang 2013).

Unfortunately, the system produced does have its limitations. The foremost of those being the BoW IR method. Not only does the BoW model discard word order and ignore context and semantics, but it is a sparse representation meaning it is significantly harder to model due to space, complexity, and information.

Another limitation, catalysed by the BoW model, is the issue of scalability. Despite this not directly effecting the problem at hand, it will have considerable impact on any future work that occurs. A delimitation was put in place around the scope of the project that the processing times would have to feasible for the size of the dataset evaluated on. However, as this project is time centric the dataset is bound to increase in size if improving the study. Alternately the classifier may be used in real time.

To summarise this project has demonstrated it has achieved the clear outlined aims and been successful in evaluating the hypothesis. However, the system produced is imperfect and there are multiple improvements that could be made to improve upon the solution. There is also a clear opportunity to take this study further and adapt its findings into new tools and systems. If the necessary improvements were made and the hypothesis was proved it its entirety, this work could eventually build to disproving the EMH and ‘Random Walk Theory’.

12 Reflection

Selecting and outlining this project, required me to consider how it I could apply my interests into an area that I would like to pursue in my future career and continued education while still challenging myself and developing new skills and knowledge. From start to finish, the project presented me with considerable challenge, other than familiarity with technical implementation elements (i.e., Python), I was previously unfamiliar with NLP and ML methods, and I was required to develop an in-depth knowledge of these topics in a short period of time. Despite this pushing myself out my comfort zone proved to be crucial to the project's success, the acquisition of knowledge drove me to constantly adapt and improve my work.

Dividing the project into sub tasks supported me in developing research and learning strategy as well as helping in dividing my time management. Dividing the project into four core areas: data collection, data processing, data analysis and data visualisation, allowed me to maintain an even distribution of recourses and effort across all aspects of the study to ensure I was developing my knowledge and understanding evenly and not focusing too heavily on an individual area.

I employed a waterfall development methodology throughout the project to structure a sequential chronological development process through all phases of the project. This method was chosen as the requirements for the project could be gathered and understood upfront. This choice of methodology also allowed a clear measured timeline to be produced.

The timeline produced was done to demonstrate what was expected to have been achieved by what date as well as the individual tasks that built towards completing a specific milestone. A timeline structure was sufficient over the implantation of a 'Gnatt Chart' as tasks could be confined to individual weeks within the project and didn't extend for longer periods of time. This proved a useful tool for time management and is something I will implement in future projects to ensure I spend an effective amount of time on tasks.

Despite this, in retrospect I should've allocated an increased period to conduct in-depth research into the fundamental concepts such as ML and NLP processing as well as the technologies that will be used to implement the project. Much of my knowledge was acquired through conducting online research, building background with academic journals, online tutorials, and studies, which proved to be a fruitful line of inquiry, however this occurred during the development process and knowledge I would've liked prior to implementation came too late.

Through implementing a double looped learning (Cartwright 2002) approach to the study I was able to think more deeply about my assumptions and beliefs towards individual aspects of the project. This allowed me to take a more fluid approach to development, changing the objectives as I discovered more about the problem at hand.

This project was of valuable insight into ML and NLP and has piqued my interest into studying these topics further. Completing a project of this scale individually, taught me valuable lessons about the development process, conducting research and

working to prove or disprove a set hypothesis. Through the research, development, and judgement of the project I have gained various skills and knowledge that will be transferable to my continued education, future career, and other projects.

13 Glossary

Aggregation – The formation of several things into a cluster.
Bearish – Characterized by or associated with falling share prices.
Bullish – Characterized by rising share prices.
Capital – Wealth in the form of money or other assets owned by a person or organization or available for a prepose such as starting a company or investing.
Causality – The relationship between cause and effect.
Conditional Independence – Situations wherein an observation is irrelevant or redundant when evaluating the certainty of a hypothesis.
Corpus – A language resource consisting of a large and structured set of texts.
Cross Validation – A resampling method that uses different portions of the data to test and train a model on different iterations.
Dimensionality – The number of input variables or features for a dataset.
Dividend – A distribution of profits by a corporation to its shareholders.
Dow Jones Industrial Average – A price-weighted measurement stock market index of 30 prominent companies listed on stock exchanges in the United States.
Feature – An individual measurable property or characteristic of a phenomenon.
Fitting – A measure of how well a machine learning model generalizes to similar data to that on which it was trained.
Gamma – Decides the level of curvature on the decision boundary.
Hedge Fund – A pooled investment fund that trades in relatively liquid assets and can make extensive use of more complex trading, portfolio-construction, and risk management techniques to improve performance.
Heterogeneous – Diverse in character or content.
Hyperparameter – A parameter whose value is used to control the learning process.
Hyperplane – A subspace whose dimension is one less than that of its ambient space.
Idiom - A group of words established by usage as having a meaning not deducible from those of the individual words.
Kernel – Takes data as input and transforms it into the required form.
Multinominal – A mathematical expression consisting of a sum of terms each of which is the product of a constant and one or more variables raised to a positive or zero integral power.

Morphological – Relating to the forms of words, in particular inflected forms.

Noise – Additional meaningless information within a dataset.

Objective – Not influenced by personal feelings or opinions in considering and representing facts.

Observations – A data point, row, or sample in a dataset. Another term for instance.

Pipeline – The end-end construct that orchestrates the flow of data into, and output from, a machine learning model.

Polysemy – The coexistence of many possible meanings for a word or phrase.

Portfolio Manager – A professional responsible for making investment decisions and carrying out investment activities on behalf of vested individuals or institutions.

S&P 500 – A stock market index tracking the performance of 500 large companies listed on stock exchanges in the United States.

Security – A fungible, negotiable financial instrument that represents some type of financial value, usually in the form of a stock, bond, or option.

Subjective – Based or influenced by personal feelings, tastes, or opinions.

Suffix – A morpheme added at the end of a word to form a derivative.

Vector – A type of array that is one dimensional.

14 Table of Abbreviations

BERT – Bidirectional Encoder Representations from Transformers

BoW – Bag-Of-Words

CSV – Comma-Separated Values

CV – Cross Validation

EDA – Exploratory Data Analysis

EMH – Efficient Market Hypothesis

GloVe – Global Vector for word representation.

IR – Information Retrieval

ML – Machine Learning

NER – Named Entity Recognition

NFLT – No Free Lunch Theory

NLP – Natural Language Processing

NLTK – Natural Language Tool Kit

POS – Part-Of-Speech

RBF – Radial Bias Function.

SCP – Simple Concordance Program

SVC – Support Vector Classification

SVM – Support Vector Machine

TF-IDF – Term Frequency-Inverse Document Frequency

UML – Unified Modelling Language

15 Bibliography

Buffett, W. 1984. The Superinvestors Graham-and-Doddsville. Available At: https://www8.gsb.columbia.edu/sites/valueinvesting/files/files/Buffett1984.pdf [Accessed: 3rd February 2022].

Cartwright, S. 2002. Double-Loop Learning: A Concept and Process for Leadership Educators What is Double-Loop Learning? Journal of Leadership Education. Available At: https://eric.ed.gov/?id=EJ1150253 [Accessed: 14th April 2022].

Devlin, J., Chang, M.-W., Lee, K., Google, K. and Language, A. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Available At: https://arxiv.org/abs/1810.04805 [Accessed: 5th February 2022].

Dodge, S., Karam L. 2017. A Study and Comparison of Human and Deep Learning Recognition Performance Under Visual Distortions. Available At: https://arxiv.org/abs/1705.02498 [Accessed: 9th April 2020].

Henriksson, J, Hultberg, C. 2019. Public Sentiment on Twitter and Stock Performance A Study in Natural Language Processing. Available At: https://www.diva-portal.org/smash/get/diva2:1354056/FULLTEXT01.pdf [Accessed: 17th February 2022].

Jakkula, V. 2006. Tutorial on Support Vector Machine (SVM). Available At: https://course.ccs.neu.edu/cs5100f11/resources/jakkula.pdf [Accessed: 7th March 2022].

Joachims, T. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. Available At: https://link.springer.com/chapter/10.1007/BFb0026683 [Accessed: 8th March 2022].

Jordan, M. I, Mitchell, T. M. 2015. Machine learning: Trends, perspectives, and prospects. Available At: https://www.science.org/doi/10.1126/science.aaa8415 [Accessed: 24th February 2022].

Law, J. 2009. A Dictionary of Business and Management (5th ed). Oxford University Press. Available At: https://www.oxfordreference.com/view/10.1093/acref/9780199234899.001.0001/acref-9780199234899 [Accessed: 10th February 2022].

Liu, Bing. 2011. Web Data Mining: Opinion Mining and Sentiment Analysis. Available At: https://link.springer.com/chapter/10.1007/978-3-642-19460-3_11 [Accessed: 6th February 2022].

Loper, E. 2022. NLTK nltk.classify.naivebayes Documentation. Available At: https://www.nltk.org/_modules/nltk/classify/naivebayes.html [Accessed: 15th March 2022].

Malkiel, B G. 2003. A Random Walk down Wall Street: The Time-Tested Strategy for Successful Investing. Available At:

https://yourknowledgedigest.files.wordpress.com/2020/04/a-random-walk-down-wall-street.pdf [Accessed: 4th February 2022].

Malo, Pekka., Sinha, Ankur; Korhonen, Pekka; Wallenius, Jyrki; Takala, Pyry. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. Journal of the Association for Information Science and Technology. Available At: https://arxiv.org/abs/1307.5336 [Accessed: 4th February 2022].

Mehta, A. 2020. Why Is Logistic Regression Called “Regression” If It Is A Classification Algorithm?. Available at: https://ai.plainenglish.io/why-is-logistic-regression-called-regression-if-it-is-a-classification-algorithm-9c2a166e7b74 [Accessed: 3rd March 2022].

Mohit, B. 2014. Named Entity Recognition. Theory and Applications of Natural Language Processing. Available At: https://link.springer.com/chapter/10.1007/978-3-642-45358-8_7 [Accessed: 12th March 2022].

Monk, A. Prins, M. Rook, D. 2019. Rethinking Alternative Data in Institutional Investment. The Journal of Financial Data Science. Available At: https://caia.org/sites/default/files/014-031_monk_jfds.pdf [Accessed: 7th February 2022].

Ni, Y., Su, Z., Wang, W. and Ying, Y. 2019. A novel stock evaluation index based on public opinion analysis. Procedia Computer Science. Available At: https://www.sciencedirect.com/science/article/pii/S1877050919302315 [Accessed: 25th February 2022].

Parveez, S. 2020. Underfitting & Overfitting—The Thwarts of Machine Learning Models’Accuracy. Available at: https://towardsai.net/p/machine-learning/underfitting-overfitting%E2%80%8A-%E2%80%8Athe-thwarts-of-machine-learning-models%E2%80%8Baccuracy [Accessed: 1st March 2022].

Patil, P. 2018. What is Exploratory Data Analysis? - Towards Data Science. Available At: https://towardsdatascience.com/exploratory-data-analysis-8fc1cb20fd15#:~:text=Exploratory%20Data%20Analysis%20refers%20to,summary%20statistics%20and%20graphical%20representations. [Accessed: 2nd March 2022].

Patro, R. 2021. Cross-Validation: K Fold vs Monte Carlo - Towards Data Science. Available at: https://towardsdatascience.com/cross-validation-k-fold-vs-monte-carlo-e54df2fc179b [Accessed: 27th February 2022].

Pennington, J. 2014. GloVe: Global Vectors for Word Representation. Available At: https://nlp.stanford.edu/projects/glove/ [Accessed: 18th April 2022].

Sammut, C. Webb, G I. 2017. Encyclopaedia of Machine Learning and Data Mining, Springer. Available At: https://link.springer.com/referencework/10.1007/978-1-4899-7687-1 [Accessed: 17th March 2022].

Sperandei, S. 2014. Understanding logistic regression analysis. Available At: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3936971/ [Accessed: 21st March 2022].

Taboada, M. 2016. Sentiment Analysis: An Overview from Linguistics. Annual Review of Linguistics. Available At: https://www.researchgate.net/publication/283954600_Sentiment_Analysis_An_Overview_from_Linguistics [Accessed: 5th February 2022].

Tavora, M. 2019. How the Mathematics of Fractals Can Help Predict Stock Markets Shifts. Available at: https://towardsdatascience.com/how-the-mathematics-of-fractals-can-help-predict-stock-markets-shifts-19fee5dd6574 [Accessed: 13th March 2022].

Uddin, M.F. 2019. Addressing Accuracy Paradox Using Enhanced Weighted Performance Metric Performance Metric in Machine Learning. 2019 Sixth HCT Information Technology Trends (ITT). Available At: https://ieeexplore.ieee.org/abstract/document/9075071?casa_token=8mroE637zh8AAAAA:xNaGZZ_1xiRv9286vrwmkUkfBKpklaSB4UjiUsUBDw8KM0Xyc4l-szEuxsfVr4queNdUiRsdXA [Accessed: 28th February 2022].

Wolpert, D. H. 2002. The Supervised Learning No-Free-Lunch Theorems. Soft Computing and Industry. Available At: https://link.springer.com/chapter/10.1007/978-1-4471-0123-9_3 [Accessed: 25th February 2022].

Zhang, C. 2013. Challenges in machine learning. Available At: https://www.researchgate.net/publication/276901525_Challenges_in_machine_learning [Accessed: 4th March 2022].