Natural Language Financial Forecasting : The South African Context

Simon Katende (KTNSIM001)

Masters Dissertation

Department of Statistical Sciences
University of Cape Town
Supervisors:
Dr Şebnem Er.
Dr Juwa Nyirenda
Dr Kanshukan Rajaratnam

Acknowledgement

Foremost, I would like to express my sincere gratitude to my supervisors Dr. Şebnem Er, Dr. Juwa Nyirenda and Dr. Kanshukan Rajaratnam for their continuous support, patience, motivation, enthusiasm and immense knowledge. Their guidance helped me in all the time of research and writing of this dissertation. I could not have imagined a better team of advisors and mentors for my study.

My family and significant other for their patience and encouragement.

Abstract

The stock market plays a fundamental role in any country's economy as it efficiently directs the flow of savings and investments of an economy in ways that advances the accumulation of capital and the production of goods and services. Factors that affect the price movement of stocks include company news and performance, macroeconomic factors, market sentiment as well as unforeseeable events. The conventional prediction approach is based on historical numerical data such as price trends and trading volumes to name a few.

This thesis reviews the literature of Natural Language Financial Forecasting (NLFF) and proposes novel implementation techniques with the use of Stock Exchange News Service (SENS) announcements to predict stock price trends with machine learning methods. Deep Learning has recently sparked interest in the data science communities, but the literature on the application of deep learning in stock prediction, especially in emerging markets like South Africa, is still limited.

In this thesis, the process of labelling announcements, the use of a more statistically relevent technique called the event study was used. Classical textual preprocessing and representation techniques were replaced with state-of-the-art sentence embeddings. Deep learning models (Deep Neural Network (DNN)) were then compared to Classical Models (Logistic Regression (LR)). These models were trained, optimized and deployed using the Tensorflow Machine Learning (ML) framework on Google Cloud AI Platform. The comparison between the performance results of the models shows that both DNN and LR have potential operational capabilites to use information dissemination as a means to assist market participants with their trading decisions.

List of Abbreviations

ADTV Average Daily Trading Volumes

ANN Artificial Neural Network

AR Auto-Regressive

ARCH Auto-Regressive Conditional Heteroskedasticity

BNS Bi-Normal Separation

BO Bayesian Optimization

BoW Bag of Words

CAR Cumulative Abnormal Return

CRF Conditional Random Fields

CSV Comma-Separated Values

DAN Deep Averaging Network

DNN Deep Neural Network

EI Expected Improvement

GP Gaussian Process

ICB Industry Classification Benchmark

Insig Insignificant Cumulative Abnormal Return (CAR)

IQR InterQuartile Range

JSE Johannesburg Stock Exchange

JSON JavaScript Object Notation

LC Learning Curves

LDA Latent Dirichlet Allocation

LR Logistic Regression

MA Moving Average

Macro avg Macro-averaged F1-score

MDA Management Discussions and Analysis

Micro avg Micro-averaged F1-score

MKSVR Multi-Kernel Support Vector Regression

ML Machine Learning

MLP Multi Layer Perceptron

MLR Multinomial Logistic Regression

MPT Modern Portfolio Theory

NLFF Natural Language Financial Forecasting

NLP Natural Language Processing

NN Neural Network

PMI Pointwise Mutual Information

PTB Penn Treebank

RBF Radial Basis Function

ReLU Rectified Linear Unit

RMSE Root Mean Squared Error

SA South Africa

SEC Securities and Exchange Commission

SENS Stock Exchange News Service

Sig - Significantly Negative CAR

Sig + Significantly Positive CAR

SVM Support Vector Machines

SVR Support Vector Regression

TF-IDF Term Frequency-Inverse Document Frequency

USA United States of America

USE Universal Sentence Encoder

Weighted avg Weighted-average F1-score

WWW World Wide Web

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Financial markets have complex structures and dynamics due to various factors such as the relationship among various stocks and sectors, investors' herding behaviour and reaction of investors to financial news. These complex dynamics have gained the attention of researchers from various domains including Finance, Mathematics, Computer Science and Data Science.

The problem of stock price prediction has been addressed since the inception of Modern Portfolio Theory (MPT) (Markowitz 1952). Initially, historical stock price data was used for stock price prediction with the help of statistical models such as the Auto-Regressive (AR), Moving Average (MA) models to more advanced models like Auto-Regressive Conditional Heteroskedasticity (ARCH) models (Schumann and Lohrbach 1993). The development of neural network models (Schumann and Lohrbach 1993) in early 90's and their ability to detect patterns in data led to the wide usage of a few other machine learning algorithms including Support Vector Regression (SVR), Fuzzy Neural Network and decomposition based models (Jothimani et al. 2017) to predict stock prices.

With the emergence of the 2nd generation World Wide Web (WWW) and the

popularity of social media, analysis of textual content in the form of news articles, blogs and other social media content such as Twitter have gained attention (Xing, Cambria, and Welsch 2018). This led to advances in Natural Language Processing (NLP), ultimately, resulted in the development of a new discipline known as Natural Language Financial Forecasting (NLFF) (Xing, Cambria, and Welsch 2018).

The importance of conducting NLFF research is vital across all markets across the globe and not just in the United States of America (USA). The need to better understand the market dynamics in emerging countries such as South Africa (SA) is imperative for the benefit of all the stakeholders of the financial markets.

1.2 Thesis Goals

In this thesis, we aim to:

1. Better understand the relationship between information dissemination and stock price trends in the SA markets and
2. utilise the relationship to forecast price changes with the eventual goal of assisting market participants in their trading decisions.

1.3 Thesis Structure

The rest of this thesis is structured as follows:

- **Chapter 2: Literature Review**
  In chapter 2, the body of literature pertaining to NLFF is reviewed from the perspective of the data sets sourced, the feature engineering applied and the machine learning models used.
- **Chapter 3: Data Pre-processing and Descriptive Analysis**
  In chapter 3, the data preprocessing steps are discussed. This involves

the process of acquiring the textual and market data, the transformation, labelling and joining of the final dataset. This is followed by descriptive analysis process where the aim is to identify relationships outlined in the thesis goals (section 1.2).

- **Chapter 4: Machine Learning Models**
  In chapter 4, the theoretical background of the machine learning algorithms applied in this thesis are discussed as well as the method used to optimize these models.

- **Chapter 5: Implementation**
  In chapter 5, the implementation of the theoretical underpinnings in chapter 4 are discussed. Beginning with the overall solution architecture, followed by an analysis of the various components of the solution architecture.

- **Chapter 6: Experiment Results and Analysis**
  In chapter 6, the experiments conducted to fulfil the goals outlined in section 1.2 are introduced and the results of the experiments are presented. The purpose of the experiments are: 1) to compare the performance of the models under different class imbalance training techniques. 2) compare the performance of Deep Learning models and Classical models in predicting price trends.

- **Chapter 7: Conclusion and Future Works**
  In this final chapter, the contributions of this thesis are reviewed and the corresponding theoretical analysis are summarised. To conclude this thesis, the limitations and the possible avenues to continue this thesis are discussed.

Chapter 2

Literature Review

Text Mining is the process of converting textual information into meaningful numerical information. Text mining techniques have been applied by many researchers in the finance area in many different ways such as predicting the stock market reaction or stock price changes to the textual financial news. In this dissertation, our aim is to analyse the effect of corporation based information on the reaction of the South African stock market. In the following section, we will summarise the relevant research done in the literature.

The research done with the aim of forecasting can be summarised according to :

i the data set used

ii the feature processing applied to the data set and

iii the machine learning models used in the analysis

The review structure is illustrated in Figure 2.1. In the following subsections, we will look into these in detail.

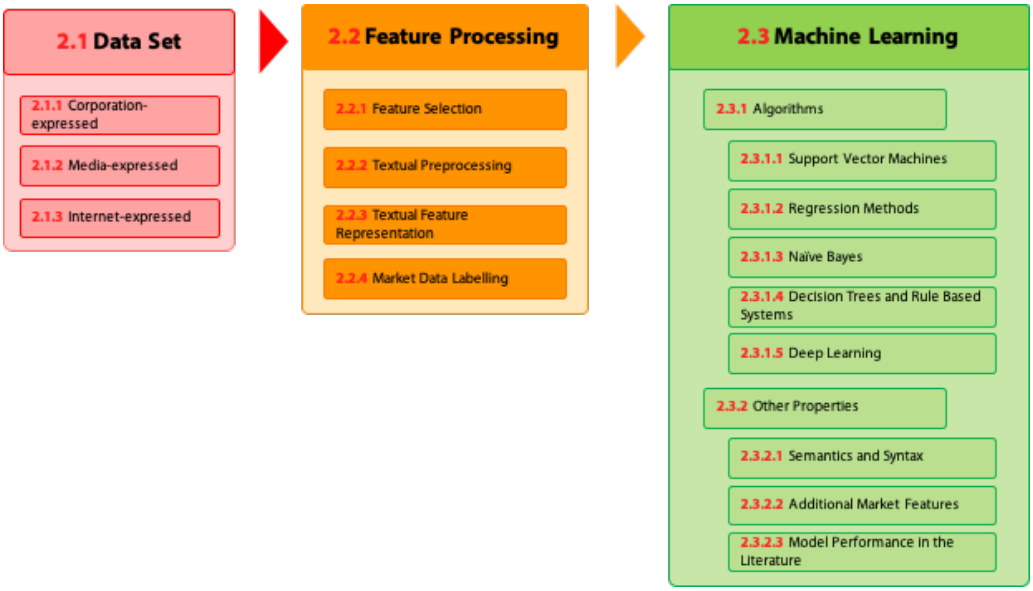

Figure 2.1: Structure of literature review

2.1 Data Set

The research in the literature uses several different types of textual data sets that can be categorised as the following (Kearney and Liu 2014) :

i Corporation-expressed

ii Media-expressed

iii Internet-expressed

2.1.1 Corporation-expressed

Corporation-expressed information is obtained from companies' annual reports, earnings releases and conference calls (Loughran and McDonald 2011, F. Li et al. 2006, Henry 2008). This growing interest in the analysis of company-related narratives is partly due to the compulsory requirements of mature financial market regulatory bodies ,such as the Securities and Exchange Commission (SEC) in the USA, for electronic filings of all listed companies in their respective markets. Two types of filings have been of

particular interest to researchers namely: the 8-K and the 10-K/Q.
The 8-K filings (or forms) provide information about significant changes in the firm whilst the 10-K/Q filings provide audited financial statements and a comprehensive overview of the companies financial condition. The body of research includes (F. Li et al. 2006, Feldman et al. 2008, F. Li 2009, Loughran and McDonald 2011, Lee et al. 2014, Henry and Leone 2015). F. Li et al. 2006 measures the risk sentiment of annual reports by counting the frequency of words related to risk or uncertainty in the 10-K filings. Feldman et al. 2008 explores whether the Management Discussions and Analysis (MDA) section of Form 10-K/Q has incremental information content beyond financial measures such as earnings surprises and accruals. F. Li 2009 examines the information content of the forward-looking statements in the MDA of 10-K/Q filings using a Naive Bayesian machine. Loughran and McDonald 2011 use 10-K flilings to develop negative word list, that better reflect tone in financial text and link the word lists to returns, trading volume and unexpected earnings. Lee et al. 2014 introduce a system that forecasts companies' stock price changes in response to financial events reported in 8-K documents. Henry and Leone 2015 evaluates alternative measures of the tone of financial narrative in the MDA of 10-K/Q filings factoring the lessons learnt from F. Li 2009 and Loughran and McDonald 2011 studies. Other work in regulatory disclosures outside of the US include (Kraus and Feuerriegel 2017, Feuerriegel and Gordon 2018) that look at the ad hoc regulatory disclosures in Germany. In the context of South Africa, the Johannesburg Stock Exchange (JSE) has the Stock Exchange News Service (SENS) which is a market dissemination service that encapsulates the above mentioned SEC reports as well as other material information types (*Announcement Types and Announcement Sub Types* n.d.). This source of information has sparked some research, however not from a NLFF perspective. The body of literature still lays a foundation to implement NLFF. The body of literature (Bhana 1998, Bhana 1999, Henn and Smit 1997, Muller 1999, Page and Way 1992, Ward and Chris Muller 2010, Watermeyer 2011) examine the impact of information and market efficiencies on the JSE with mixed outcomes.

2.1.2 Media-expressed

Media-expressed information originates from news stories and analyst reports (Tetlock 2007). Tetlock, Saar-Tsechansky, and Macskassy 2008 studied the effect of news stories on future earnings and stock returns, showing that

the fraction of negative words in firm-related news stories predicts both low earnings and low stock returns. X. Li et al. 2014 demonstrated that news stories can be utilised to improve the accuracy of prediction on stock returns in intra-day trading. Schumaker and Chen 2009 examined a Support Vector Machines (SVM) approach for financial news articles analysis using several textual representations: Bag of Words (BoW), noun phrases, and named entities. Engelberg, Reed, and Ringgenberg 2012 showed that there is a dramatic increase in short selling after news events, providing an information advantage to informed traders. The impact of media may also vary according to firm characteristics and article content (Q. Li, TieJun Wang, et al. 2014). The majority of the sources used are major news websites such as the Wall Street Journal (Antweiler and Frank 2004) and Yahoo! Finance (Antweiler and Frank 2004, Schumaker, Y. Zhang, et al. 2012).

2.1.3 Internet-expressed

Internet-expressed information, and in particular sentiment, is used to extract the information from small investors (Antweiler and Frank 2004). For example, in their stock price prediction model, Q. Li, Tiejun Wang, et al. 2014 combined news information with the information obtained from online financial discussion boards. Similarly, Y. Yu, Duan, and Q. Cao 2013 investigated content from social media, including blogs, forums, and Twitter. Their findings suggest that social media has a stronger impact on companies stock performance than conventional media.

2.2 Feature Processing

Feature Processing is a crucial part of text mining. There are four subprocesses which we will be discussing, namely:

i Feature Selection

ii Textual Preprocessing

iii Feature Representation

iv Market Data Transformation

2.2.1 Feature Selection

The array of feature selection techniques used in the literature are summarised in Table 2.1. The popular technique used in literature is the BoW which consists of separating text into its words and considering each word as a feature. As viewed in Table 2.1 around 58% of the literature rely on this selection method of which the semantic relation between words and order is ignored (Schumaker, Y. Zhang, et al. 2012, Schumaker and Chen 2009). Some of the other works use the n-grams technique (Butler and Kešelj 2009, Hagenau, Liebmann, and Neumann 2013) which is adjoining sequence of words. This can be advanced with the introduction of syntactic structures with a particular paper that utilises syntactic n-grams (Sidorov et al. 2014). Other feature selection techniques involve the combination of lexical and semantic features for text classification (Yang et al. 2013).

2.2.2 Textual Preprocessing

Textual preprocessing approaches taken by the various works is shown in Table 2.1. The first approach which is normally the foundation of textual preprocessing before further dimensionality reduction methods occur are: features lemmatization and stemming, conversion to lowercase, removal of punctuation marks and numbers, removal of web page addresses and stopwords. In some of the literature, the abovementioned are the only textual preprocessing steps taken (Fung, J. X. Yu, and Lam 2003). The other approaches, which can also be seen as dimensionality reduction, are taking a top-N of words where N is the top number of words out of all the words (Zhai, Hsu, and Halgamuge 2007). The most common approach is setting a minimum occurrence threshold (Butler and Kešelj 2009, Schumaker and Chen 2009). Another common approach is the use of a predefined dictionary that replaces a word with a category name or a numeric value. There are various types of dictionaries in the literature: dictionaries that are compiled by market experts (Wuthrich et al. 1998, Peramunetilleke and R. K. Wong 2002), psychology related dictionaries like the Harvard-IV-4 which has been used in the following paper (Tetlock, Saar-Tsechansky, and Macskassy 2008). A more general dictionary includes the WordNet thesaurus used by (Tetlock, Saar-Tsechansky, and Macskassy 2008, Zhai, Hsu, and Halgamuge 2007). A more advanced case is when a dictionary is created dynamically based on the text corpus using a term extraction tool (Soni, Eck, and Kaymak 2007).

Finally, the last approach which can be seen as both a textual preprocessing as well a feature representation (see section 2.2.3) is word embeddings. Word embeddings is a dense vector representation of words (as opposed to one hot encoded) from a pre-trained embeddings model, examples of this are the popular Word2vec and GloVe. Word embeddings have a useful property in that words that have similar context in terms of neighbouring words tend to have vectors that are collinear. This has seldomly been used in the literature with only one paper (Huynh, Dang, and Duong 2017).

2.2.3 Textual Feature Representation

Once the features have been sufficiently reduced, each feature needs to be represented numerically in order to be ingested into a machine learning algorithm. Table 2.2 summarises the various research papers on feature representation studies. The most basic feature representation is the binary representation where two values (0 or 1) represent the absence or presence of a feature for example a word in the case of a BoW as with the following (Schumaker, Y. Zhang, et al. 2012, Wuthrich et al. 1998). The next common feature representation is the Term Frequency-Inverse Document Frequency (TF-IDF) whereby the TF-IDF increases proportionally to the occurance of the words in a document and is offset by the frequency of the word in the corpus so as not to bias popular words (Hagenau, Liebmann, and Neumann 2013, Peramunetilleke and R. K. Wong 2002, Fung, J. X. Yu, and Lam 2003, Peramunetilleke and R. K. Wong 2002).

2.2.4 Market Data Labelling

The majority of the price movement forecasting types in the body of literature are categorical with discrete values like *Up*, *Down* and *Stay*. Furthermore the price movement labelling methodology tends to be derived from simple differencing of returns with a benchmark and comparing that with a threshold to classify to the above mentioned. A few researchers utilise a more systematic, statistically relevant method called the event study that utilises the Cumulative Abnormal Return (CAR) over some event window, (Feldman et al. 2008, Henry 2008, Henry and Leone 2015, Loughran and McDonald 2011, Engelberg, Reed, and Ringgenberg 2012). The benefit of using abnormal returns is that it corrects returns for confounding market movements and isolates the effect of the disseminated information itself.

2.3 Machine Learning

In this section we will provide a brief summary of the algorithms in the literature as well as a comparison of the other detailed technicalities in the designs of the reviewed systems.

2.3.1 Algorithms

Support Vector Machines

The Support Vector Machines (SVM) algorithm is a non-probabilistic binary linear classifier used for supervised learning. The objective is to find a hyper-plane that separates two classes with a maximum margin. The training problem in SVM can be represented as a quadratic programming optimisation problem. A common implementation method of SVM in the literature is the SVM light (Mittermayer 2004, Fung, J. X. Yu, and Lam 2003). The SVM Light (Joachims 2002) is an implementation of an SVM learner which addresses the problem of large tasks. SVM can be extended to nonlinear classifiers by applying kernel mapping (also known as kernel trick). As a result of applying the kernel mapping, the original classification problem is transformed into a higher dimensional space. The kernel function used may influence the performance of the classifier. Zhai, Hsu, and Halgamuge 2007 utilise both Gaussian Radial Basis Function (RBF) and polynomial kernels in their research.

Regression Methods

Regression methods take on numerous forms in the literature as viewed on Table 2.2. The first type of regression is the SVR which is a regression equivalent of SVM but without the aspect of classification (Vapnik 2013). This method has been used by (Schumaker, Y. Zhang, et al. 2012 , Hagenau, Liebmann, and Neumann 2013, Tay and L. Cao 2001, Schumaker and Chen 2009). Hagenau, Liebmann, and Neumann 2013 use SVR to assess the ability to predict the discrete value of the stock return.

Linear Regression models are also used. Tetlock, Saar-Tsechansky, and Macskassy 2008 use two different dependent variables (raw and abnormal next

day returns) regressed on different negative words measures. Their main result is that negative words in firm-specific news stories robustly predict slightly lower returns on the following trading day. Jin et al. 2013 applies topic modelling methods and uses customised sentiment dictionaries to uncover sentiment trends by analysing relevant sentences. The linear regression model then estimates the weight for each topic and makes currency forecasts.

Naive Bayes

Naive Bayes is quite popular in the literature (Groth and Muntermann 2011, F. Li 2010, Antweiler and Frank 2004, Wuthrich et al. 1998). Naive Bayes is based on the Bayes Theorem and is called "Naive" based on the assumption that the occurrences of words are independent of each other - which is a highly unrealistic assumption. This algorithm differentiates itself from the other models (e.g SVM) in that it is built upon probabilities (of a feature belonging to a category) whereas the other algorithms interpret the document feature-matrix spatially. F. Li 2010 uses Naive Bayes to examine the information content of the forward looking statements in the MDA section of corporate filings.

Decision Trees and Rule Based Systems

A few researchers have made efforts to utilise rule and decision tree based classification systems (Huang et al. 2010, Peramunetilleke and R. K. Wong 2002, Vu et al. 2012, Rachlin et al. 2007). Peramunetilleke and R. K. Wong 2002 use a set of keywords provided by financial domain experts. The classifier expressing the correlation between the keywords and one of the outcomes is a rule set. Each of the rule sets ($dollar\_down$, $dollar\_steady$, $dollar\_up$) yields a probability of how likely the respective event will occur in relation to available keywords. Huang et al. 2010 observes that the combination of keywords in financial news headlines plays a crucial role on the next trading day. They then applied weighted association rules algorithm to detect the important compound terms in the news headlines. Rachlin et al. 2007 use the decision trees algorithm, and in particular, the C4.5 algorithm developed by Quinlan 2014. This algorithm produces a set of trend (Up, $Down$, $Expected$) predicting rules. They also show the effect of the combination between numerical and textual data, what they found, contrary to our expectations, is that the results did not indicate that the textual information can improve

the predictive accuracy of the numeric analysis. Vu et al. 2012 also uses the C4.5 decision tree for classifying daily up and down changes in stock prices. From the rule based decision categorisers, the rules are composed of a set of words which can be insightful. More than just attempting to assign a label, a set of decision rules may summarise how to make decisions. For example, the rules may suggest a pattern of words found in the news prior to the rise of a stock price. The downside of rules is that they can be less predictive if the underlying concept is complex (Weiss, Indurkhya, and T. Zhang 2015).

Deep Learning

Deep Learning has rarely featured in the literature. The few instances are (Peng and Jiang 2015, Kraus and Feuerriegel 2017, Huynh, Dang, and Duong 2017). Peng and Jiang 2015 apply the popular word embedding methods and deep neural networks to predict stock price movements in the market based on financial news. Their experiments have shown that the financial news is very useful in stock prediction and the proposed methods can significantly improve the prediction accuracy on a standard financial data set.
Kraus and Feuerriegel 2017 studies the use of deep neural networks for financial decision support. They additionally experiment with transfer learning, in which they pre-train the network on a different corpus with a length of 139.1 million words. Their results reveal a higher directional accuracy as compared to traditional machine learning when predicting stock price movements in response to financial disclosures.

2.3.2 Other Properties

There are additional properties over and above the discussed that one needs to observe when looking at the Machine Learning algorithms used for NLFF. This is summarised in Table 2.2 and is explained in the upcoming sections.

Semantics and Syntax

Semantics deals with the meaning of words whilst syntax deals with their order and relative positioning. As observed in Table 2.2, about 53% of the research utilise some semantic aspect in their text mining approach which is usually done by using a dictionary or thesaurus and categorising the words based on their meaning. Few in the literature have made an effort to include

syntax. The basic approaches include noun-phrases (Schumaker and Chen 2009), word-combinations and n-grams (Schumaker and Chen 2009, Hagenau, Liebmann, and Neumann 2013) as well as simultaneous appearance of words (Huang et al. 2010).

Additional Market Features

Additional market data or signals along with the text features can be included into the classification algorithm as features. Examples could be price or index levels and/or average daily trade volume. Technical signals are the outputs of technical algorithms or rules for example the moving average as well as the relative strength index. Few in the literature have taken the advantage of the additional features.(Butler and Kešelj 2009, Butler and Kešelj 2009, Hagenau, Liebmann, and Neumann 2013, Schumaker and Chen 2009, Schumaker, Y. Zhang, et al. 2012).

Model Performance in the Literature

Majority of the categorical models in the literature present their results in the form of a confusion matrix. Accuracy, the commonly used metric, achieved a result range of 50%-70% (Schumaker, Y. Zhang, et al. 2012, Butler and Kešelj 2009, F. Li 2010, Schumaker and Chen 2009, Schumaker, Y. Zhang, et al. 2012, Zhai, Hsu, and Halgamuge 2007). What makes the results and the choice of performance metric controversial is that the majority of the literature have not reported whether their test data class labels is imbalanced or not. Among the reviewed works, only three papers (Soni, Eck, and Kaymak 2007, Mittermayer 2004, Peramunetilleke and R. K. Wong 2002) have paid some attention to this topic in their works.
Another evaluation approach is the implementation of a trading strategy (Tetlock, Saar-Tsechansky, and Macskassy 2008, Schumaker and Chen 2009, Schumaker, Y. Zhang, et al. 2012, Hagenau, Liebmann, and Neumann 2013, Fung, J. X. Yu, and Lam 2003, Zhai, Hsu, and Halgamuge 2007, Mittermayer 2004, Groth and Muntermann 2011, Huang et al. 2010, Rachlin et al. 2007) through which a trading period is simulated and profits are measured to evaluate the viability of the system. In general researchers are using evaluation mechanisms and experimental data that widely vary and this makes an objective comparison in terms of concrete levels of effectiveness unreachable.

Tables 2.1 and 2.2 below provide a summary of the literature reviewed.

Reference	Year	Text type	Text Source	No of items	Frequency	Period	Forecast Type	Feature Selection	Textual Preprocessing
Fung, J. X. Yu, and Lam 2003	2003	Media-Expressed	Reuters	600000	Daily	1 October 2002 – 30 April 2003	Categorical: *up*, *down*	BoW	Lemmatization
Antweiler and Frank 2004	2004	Internet-Expressed	Yahoo Finance, Raging Bull, Wall Street Journal	1500000 messages	Intraday	1 January – 31 December 2000	Categorical: *buy*, *sell*, *hold*	BoW	Top N words
Mittermayer 2004	2004	Media-Expressed	Not mentioned	6600	Daily	1 Jan – 31 Dec 2002	Categorical: *goodnews*, *badnews*, *nomovers*	BoW	Top N words
Rachlin et al. 2007	2007	Media-Expressed	Forbes.com, Reuters.com	No mentioned	Days	7 Feb 2006 - 7 May 2006	Categorical: *up*, *slight – up*, *expected*, *slight–down*, *down*	BoW, commonly used financial values	Most influential keywords list
Soni, Eck, and Kaymak 2007	2007	Media-Expressed	Financial Times Intelligence	3493	Daily	1 Jan 1995 - 15 May 2006	Categorical: *positive*, *negative*	Visualisation	Thesaurus made from term extraction tool
Tetlock, Saar-Tsechansky, and Macskassy 2008	2007	Media-Expressed	Wall Street Journal,Dow Jones News	350000	Daily	1980 - 2004	Regression	BoW	predefined dictionary. Harvard-IV-4 psycho-social dictionary.
Butler and Kešelj 2009	2009	Corporate-Expressed	Company websites	Not mentioned	Yearly	2003 - 2008	Categorical: *Over−*, *Under−* Perform index	Character n-Grams, three readability scores, last year's performance	Minimum occurrence per document
F. Li 2009	2009	Corporate-Expressed	MDA section of 10-K and 10-Q filings from SEC Edgar Web site	13 million forward-looking statements in 140 000 10-Q and K filings	Annually	1994 - 2007	Categorical: *positive*, *negative*, *neutral*, *uncertain*	BoW, tone and content	Predefined dictionaries
Schumaker and Chen 2009	2009	Corporate-Expressed	MDA section of 10-K and 10-Q filings from SEC Edgar Web site	13 million forward-looking statements in 140,000 10-Q and K filings	Intraday	26 Oct 2005 –28 Nov 2005	Categorical: discrete numeric	BoW, noun phrases, named entities	Minimum occurrence per document
Huang et al. 2010	2010	Media-Expressed	Leading electronic newspapers in Taiwan	12830 headlines	Daily	Jun 2005 – Nov 2005	significance degree assignment	Simultaneous terms, ordered pairs	Synonyms replacement
Groth and Muntermann 2011	2011	Corporate-Expressed	Corporate disclosures	423	Intraday	1 Aug 2003 – 31 Jul 2005	Categorical: *positive*, *negative*	BoW	Feature scoring methods using both Information Gain and Chi-Squared metrics
Hagenau, Liebmann, and Neumann 2013	2013	Corporate-Expressed	DGAP and Euro Adhoc	10870 DGAP announcements and 3478 Euro Adhoc announcements	Daily	1997 - 2001	Categorical: *positive*, *negative*	BoW, noun phrases, word combinations, n-grams	Frequency for news, Chi2-approach and Bi-Normal Separation (BNS) for exogenous-feedback-based feature selection, dictionary
Schumaker, Y. Zhang, et al. 2012	2012	Media-Expressed	Yahoo Finance	2802	Intraday	26 Oct 2005 - 28 Nov 2005	Regression	Opinion Finder overall tone and polarity	Minimum occurrence per document
Vu et al. 2012	2012	Internet-Expressed	Twitter	5001460	Daily	1 Apr 2011 – 31 May 2011, online test 8 Sept –26 Sept 2012	Categorical: *up*, *down*	Daily aggregate number of positives or negatives on Twitter and an emoticon lexicon. Daily mean of Pointwise Mutual Information (PMI) for predefined bullish-bearish anchor words	Predefined company related keywords, Named Entity Recognition based on linear Conditional Random Fields (CRF)
Jin et al. 2013	2013	Media-Expressed	Bloomberg	361782	Daily	1 Jan 2012 – 31 Dec 2012	Regression	Latent Dirichlet Allocation (LDA)	Topic extraction, top topic identification by manually aligning news articles with currency fluctuations
Lee et al. 2014	2014	Corporate-Expressed	SEC Edgar Web site	13671 8K filings	Daily	2002 - 2012	Categorical: *up*, *down*, *stay*	N-grams	Lemmatization, minimum occurrence of unigrams
Q. Li, Tiejun Wang, et al. 2014	2014	Media-Expressed Internet Expressed	sina.com eastmoney.com finance.caixin.com finance.people.com.cn	124470	Intraday	1 Jan 2011 - 31 Dec 2011	Regression	BoW, noun phrases	Predefined dictionaries
X. Li et al. 2014	2014	Media-Expressed	Caihua finet.hk	28885	Intraday	2001	Regression	BoW	Chi-squared
Peng and Jiang 2015	2015	Media-Expressed	Reuters and Bloomberg	106521 and 447145 respectively	Daily	1 Oct 2006 - 31 Dec 2013	Categorical: up, down	Bag of Keywords	Top N words
Huynh, Dang, and Duong 2017	2017	Media-Expressed	Reuters and Bloomberg	106521 and 447145 respectively	Daily	1 Oct 2006 - 31 Dec 2013	Categorical: *up*, *down*	BoW	Word Embedding

Table 2.1: Table of the literature discussed 1

Reference	Year	Feature Representation	Algorithm Type	Semantics	Syntax	Financial Features	Findings	Trading strategy
Peramunetilleke and R. K. Wong 2002	2002	Boolean, TF-IDF, TF-CDF	Decision Rules or Trees	Yes	No	No	Better than chance	No
Fung, J. X. Yu, and Lam 2003	2003	TF-IDF	SVM	No	No	No	The cumulative profit of monitoring multiple time series is nearly double to that of monitoring single time series.	Yes
Antweiler and Frank 2004	2004	Binary	Naïve Bayes, SVM	No	No	No	Evidence that the stock messages help predict market volatility, but not stock returns	No
Mittermayer 2004	2004	TF-IDF	SVM	No	No	No	Average profit 11% compared to average profit by random trader 0%	Yes
Rachlin et al. 2007	2007	TF,Boolean	C4.5 decision tree	No	No	Yes	Cannot improve the predictive accuracy of the numeric analysis. Accuracy 82.4%for join textual and numeric analysis and 80.6% for textual analysis, and 83.3% numeric alone	Yes
Soni, Eck, and Kaymak 2007	2007	Visual coordinates	SVM	Yes	No	No	Hit rate of classifier: 56.2% compared to 47.5% for naïve classifier and 49.1% SVM BoW	No
Tetlock, Saar-Tsechansky, and Macskassy 2008	2007	Frequency divided by total number of words	Regression	Yes	Yes	No	(1) The fraction of negative words in firm-specific news stories forecasts low firm earnings; (2) firms' stock prices briefly under-react to the information embedded in negative words; and (3) the earnings and return predictability from negative words is largest for the stories that focus on fundamentals	Yes
Butler and Kešelj 2009	2009	Frequency of the n-gram in one profile	CNG distance measure & SVM & combined	No	No	Yes	First method: 55% and 59% for character-grams and word-grams accuracy respectively, still superior to the benchmark portfolio. Second method: overall accuracy and over-performance precision was 62.81% and 67.80% respectively	No
F. Li 2009	2009	Binary, Dictionary value	Naïve Bayes and dictionary-based	No	No	No	Accuracy for tone 67% and content 63% with Naïve Bayes and less than 50% with dictionary-based	No
Schumaker and Chen 2009	2009	Binary	SVM	Yes	Yes	Yes	Directional accuracy 57.1%, return 2.06%, closeness 0.04261	Yes
Huang et al. 2010	2010	Weighted based on the rise/fall ratio of index	Weighted association rules	Yes	Yes	No	Prediction accuracy and the recall rate up to 85.2689% and 75.3782% in average,respectively	Yes
Groth and Muntermann 2011	2011	TF-IDF	Naïve Bayes, k-NN,ANN, SVM	No	No	No	Accuracy (slightly) above the 75% guessing equivalent benchmark	Yes
Hagenau, Liebmann, and Neumann 2013	2013	TF-IDF	SVM with a linear kernel, SVR	Yes	Yes	Yes	Feedback-based feature selection combined with 2-word combinations achieved accuracies of up to 76%	Yes
Schumaker, Y. Zhang, et al. 2012	2012	Binary	SVR	Yes	No	Yes	Objective articles were performing poorly in directional accuracy vs. baseline. Neutral articles had poorer trading returns vs. baseline. Subjective articles performed better with 59.0% directional accuracy and a 3.30% trading return	Yes
Vu et al. 2012	2012	Real number for Daily Neg_Pos and Bullish_Bearish	C4.5 Decision tree	Yes	Yes	No	Combination of previous days's price movement, bullish/bearish and Pos_Neg features create a superior model in all 4 companies with accuracies of: 82.93%,80.49%, 75.61% and 75.00% and for the online test as: 76.92%, 76.92%, 69.23% and 84.62%	No
Jin et al. 2013	2013	Each article's topic distribution	Linear regression model	Yes	No	No	Precision around 0.28 on average	No
Lee et al. 2014	2014	Binary	Random Forest	No	No	Yes	Results indicate that using text boosts prediction accuracy over 10% (relative) over a strong baseline that incorporates many financially-rooted features	
Q. Li, Tiejun Wang, et al. 2014	2014	TF-IDF	SVR	Yes	No	No	Experiments on the CSI 100 stocks during a three-month period show that a predictive performance is 0.612 in terms of Root Mean Squared Error (RMSE), the same direction of price movement as the future price is 55.08%, and a simulation trading return is up to 166.11%	Yes
X. Li et al. 2014	2014	TF-IDF	Multi-Kernel Support Vector Regression (MKSVR)	No	No	No	Results show that the MKSVR is capable of making the better use of hidden information in news articles and historical stock prices than the models that simply use news articles to forecast stock prices.	No
Huynh, Dang, and Duong 2017	2017	Word Vectors	Deep Learning Models (BGRU)	No	Yes	No	Accuracy of nearly 60% in S%P 500 index prediction and over 65% for individual stock prediction.	No
Kraus and Feuerriegel 2017	2017	Word Vectors	Deep Learning Models (RNN, LSTM)	No	Yes	No	LSTM models can outperform all traditional machine learning models based on the BoW approach, especially when we further pre-train word embeddings with transfer learning.	No

Table 2.2: Table of the literature discussed 2

Chapter 3

Data Preprocessing and Descriptive Analysis

This chapter introduces the process involved in acquiring the textual and market data sets, the steps taken in transforming, labelling and finally joining the data sets. Descriptive analysis was then conducted on the joined dataset. We conclude the chapter by discussing the training methods needed to address the issues that arise from labelling the data.

3.1 Data Sets

There are primarily two types of data sets in this project, namely:

i Textual Data

ii Market Data

3.1.1 Textual Data

Textual Data Acquisition

All available SENS announcements from August 2012 through to January 2019 were obtained from Moneyweb, using a customized web scraping algorithm on **R** leveraging off the *Rvest* library.

Market Liquidity Filtration

The scraped announcements, or events are used interchangeably throughout this thesis, were further filtered by observing currently listed companies as at 8 February 2019 obtained from Bloomberg. Corporate debt issuance and delisted companies' announcements were excluded from the data set. This then brought the total announcement size to 36792. We restrict the data set further by applying a liquidity filter by applying the following rules :

i Average Daily Trading Volumes (ADTV) (30 days)[1] greater than 1000 were included.

ii ADTV (1 Year)[1] greater than 1000 were included.

iii Market Capitalisation [1] greater than or equal to R 1 billion were included.

iv Share Turnover [2] greater than or equal to 0.01 were included.

This then brought the total announcement size to 19861. The above restrictions are due to liquidity required for effective trading, as well as for effectively deriving market model parameters, which will be discussed in detail in Section 3.2.

[1] as at 15 July 2019

[2] $ShareTurnover = \frac{AverageTradingVolume(1Year)}{SharesOutstanding(Float)} \times 1000$

Textual Feature Extraction

The following features are extracted from the raw announcement text through the use of regular expressions leveraging off the *stringr* library on **R**:

Feature Name	Data Type	Description
datetime	date time	Timestamp of the announcement
ticker	string	Stock identification code
sens_headline	string	Headline of the announcement
sens_content	string	Full details of the announcement
announcement_type	string	Lemmatized sens_headline

Table 3.1: Summary of all the features in the text dataset

The *announcement_type* feature is derived from the *sens_headline* feature by removing the company specific information and conducting lemmatization. Figure 3.1 below is the textual extraction process.

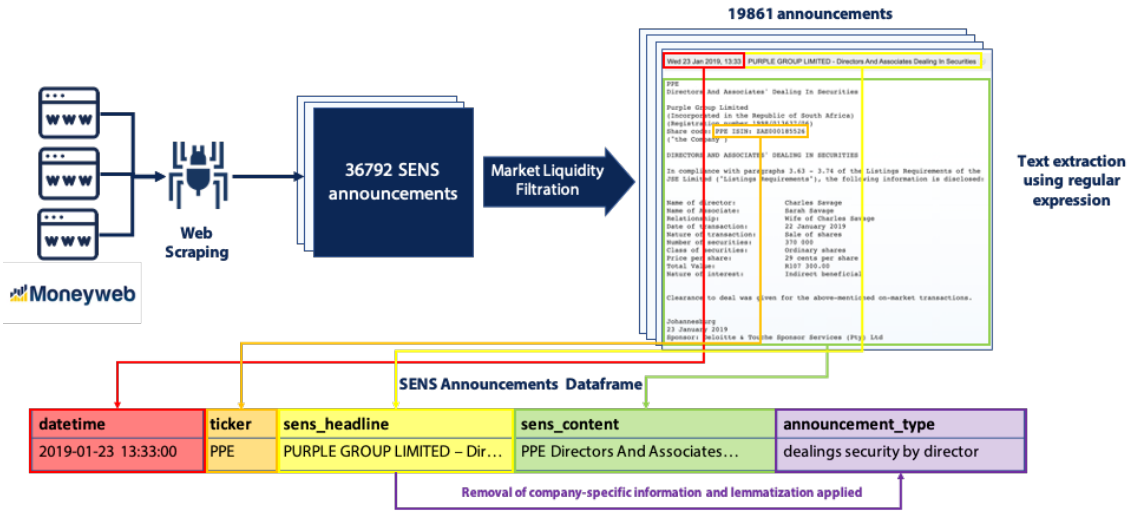

Figure 3.1: SENS announcements extraction process

Textual Data Representation

As discussed in Section 2.2.3, traditionally textual representation has been in the form of one hot encoding a text corpus created from methods like BoW and TF-IDF. The challenge of using such a representation is that it creates high dimensional vectors to account for all the words in a corpus with 99% of the elements in each vector as zero.
A more efficient method that reduces the curse of dimensionality, whilst factoring the similarity of words. This method is called word embeddings.
Word embeddings are a dense vector representation of floating point values. Instead of assigning values to these vectors arbitrarily, these values are semantically determined from pre-trained models that are summarised as either frequency based (example Glove (Pennington, Socher, and Manning 2014)) or prediction based (Word2Vec (Mikolov et al. 2013)).
What is implemented in this thesis is an advancement of word embeddings, which is sentence embeddings, whereby the sentence encoder ingests the text as is and encodes it to a semantically-meaningful fixed length dense vector. The particular sentence embedding implemented is the Universal Sentence Encoder (USE) developed by Google which in essence is a deep learning model pre trained on a large corpus (Cer et al. 2018). In particular, the encoding model makes use of a Deep Averaging Network (DAN) (Iyyer et al. 2015). How the USE model is implemented on TF-Hub are as follows:

1. The sentences are pre-processed via textual-normalisation (lowercasing, stemming, lemmatization, removal of stop words). This is then followed by tokenization into tokens using the Penn Treebank (PTB) tokenizer[3].

2. The tokenized words as well as their bi-grams are then embedded and averaged together.

3. The averaged vectors are passed through a pre-trained 4-layer feed-forward DNN to get 512-dimensional sentence embedding as an output.

Figure 3.2 below is an illustration of the USE model as explained in the steps above.

[3] https://web.archive.org/web/19970714014358if_/http://www.cis.upenn.edu/~treebank/tokenizer.sed

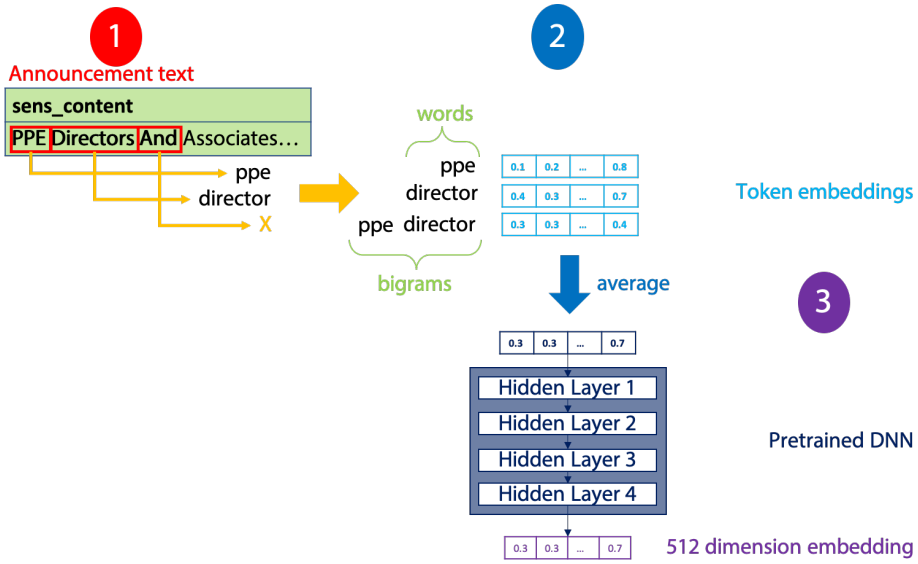

Figure 3.2: Sentence embedding process

The rational behind the use of a DAN is that each layer will increasingly magnify small but meaningful differences in the word embedding average. In addition, it proves to be computationally efficient as opposed to its Transformer based encoder counterpart (Cer et al. 2018).

3.1.2 Market Data

Market Data for the corresponding stocks were extracted from Bloomberg and Yahoo Finance using the *Rblpapi* and *tidyquant* **R** packages respectively. Market data was extracted with at least 120 trading days before the first announcement for event study purposes. Table 3.2 below summarises the features:

Feature Name	**Data Type**	**Description**
ticker	string	Stock identification code
date	datetime	Date
open	float	Opening price of stock
high	float	Highest price the stock went on that particular trading day
close	float	Closing price of the stock on that particular trading day
volume	integer	Volume traded on that particular trading day
adjusted	float	Adjusted closing price

Table 3.2: Summary of all the features in the market dataset

In addition to the above-mentioned daily market data, the market capitalisation and the ADTV for 30 days and 1 year for all the respective stocks were taken as at 15 July 2019 as a potential feature as well as for filtering out illiquid stocks as discussed in Section 3.1.1.

3.1.3 Textual Data Distribution

Below is a graph of the distribution of SENS announcements grouped by months.

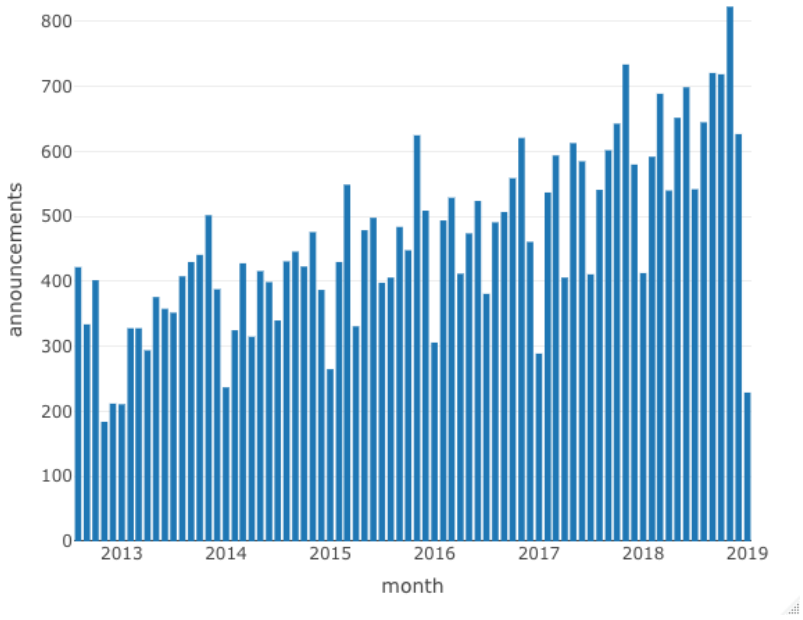

Figure 3.3: SENS announcements grouped by months

From the illustration in Figure 3.3 above we can see that there is a growing trend of announcements as the years go by. Secondly, within any particular year one can observe an oscillating pattern whereby the announcements tend to ramp up until November where it peaks and December it tends to drop.

3.1.4 Announcement Distribution by Announcement Type

Figure 3.4 is a graph of the distribution of SENS announcements grouped by announcement type. Given that there was lack of consistency with their announcement type naming, a process of lemmatization was conducted to normalise the nuanced words used as a means to further classify the announcement types.

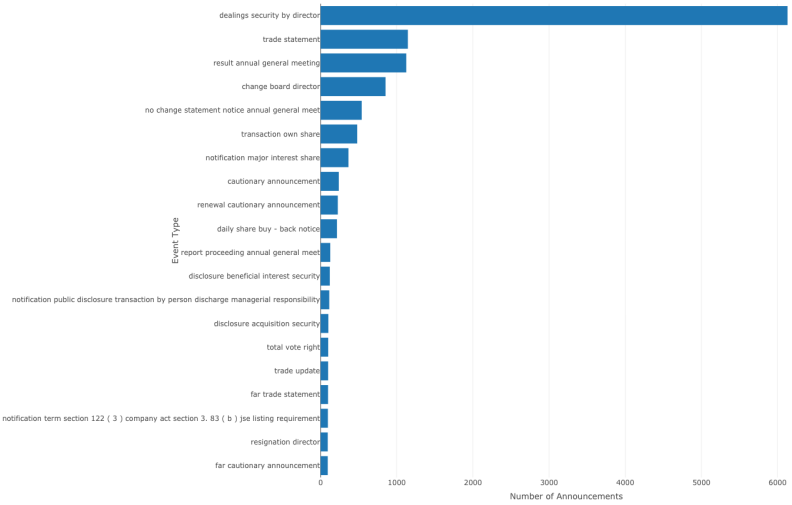

Figure 3.4: SENS announcements grouped by announcement type

From the illustration in Figure 3.4 above we can see that the announcements are heavily dominated by *dealings from directors* which makes sense as their share options vest, they then sell them and by law they have to disclose such activities to the market. The other observation made, which coincides with what was observed in the Watermeyer 2011 paper, is that there is a large volume of "noise" announcements which tends to explain the diluted announcement types which could negatively impact the effectiveness of this feature for model prediction.

3.2 Event Study

In finance theory, a semi-strong efficient market reflects all available information about firms in their stock prices. Given this basic premise, one can study how a particular event changes a firm's prospects by quantifying the impact of the event on the firm's stock. We quantify the impact of an event by computing the abnormal returns. Abnormal returns, within an event window, are computed by subtracting the returns that would have been realised (normal returns) if the analysed event did not take place from the actual returns of the stock($R_{i,t}$). Whilst the actual returns are obtained empirically, the normal returns are estimated by making use of numerous expected returns models. The most frequently used expected return model is the market model; which looks at the actual returns of a reference market (in our case the JSE All Share Index $R_{m,t}$) and the correlation of the firm's stock with the reference (expressed by the α_i and β_i parameters) which is computed on the estimation period, which in the case of this thesis is 120 days. This is illustrated in Equation 3.1 and Figure 3.5 below:

$$AR_{i,t} = R_{i,t} - (\alpha_i + \beta_i R_{m,t}) \tag{3.1}$$

In order to determine the total impact of an event especially in the event that the markets respond either earlier than the event day ($t = 0$) due to potentially acting on inside information or later than the event day, we need to observe the period around the event day termed the event window. We will then add up the individual abnormal returns in the event window to create the Cumulative Abnormal Return (CAR) as illustrated in Equation 3.2. The event window chosen was a period of 5 days starting at $t_1 = -2$ and ending at $t_2 = 2$.

$$CAR(t_1, t_2) = \sum_{t=t_1}^{t_2} AR_{i,t} \tag{3.2}$$

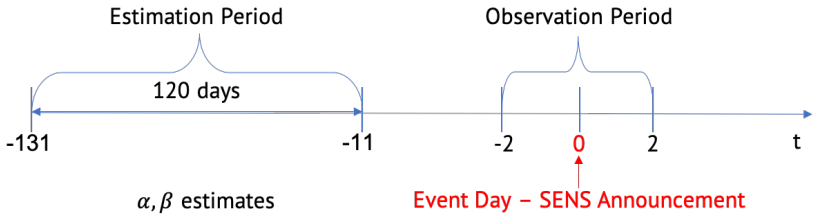

Figure 3.5: Events study timeline

3.2.1 Significance Testing

In order to determine what event constitutes an abnormal event, one needs to observe the abnormalities relative to the other events in the market.
In the literature, many efforts were conducted looking at both parametric and non parametric methods (Cowan 1992) focusing mainly at a sample of events. This proves to be not applicable in the case of this thesis as we are looking at individual events for the purposes of labelling individual events as a means to conduct supervised learning.
The alternative method conducted is to look at the distribution of all the CAR across all stocks in the dataset, which is illustrated in Figure 3.6 below

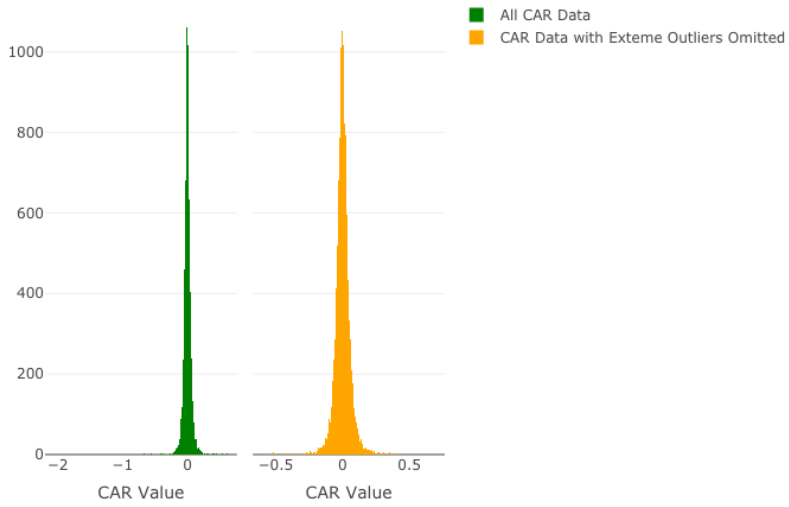

Figure 3.6: CAR distributions with extreme outliers (left) and without (right)

From the illustration in Figure 3.6 above on the leftmost side, there appears

to be extreme outlier events that are skewing the distribution to the left. The statistical summary of the data with and without the extreme outliers are illustrated in the table below

	With Extreme Outliers	**Without Extreme Outliers**	**Remarks**
Observations	19 861	19856	5 less observations
Mean	0.11%	0.15%	Shifted to the right
Variance	0.44	0.38	Smaller variance
Skewness	-3.389	0.256785	Far less skewed
Kurtosis	111.69	17.87892	Decreased Kurtosis

Table 3.3: Statistical summary data with and without extreme outliers

Below is a summary of the extreme outlier events.

No	Event Date	Company Name	SENS Title	CAR %
1	08.12.2017	Steinhoff International Holdings NV	Steinhoff Reschedules Meeting With Bankers To 19 December 2017	-132,07
2	07.12.2017	Steinhoff International Holdings NV	Ad Hoc: Steinhoff Update On Market Concerns Following Delay In Audited Results Due To Further Investigations Require	-164,32
3	06.12.2017	Steinhoff International Holdings NV	Steinhoff Announces Investigation Into Accounting Irregularities And Resignation Of CEO	-219,16
4	04.12.2017	Steinhoff International Holdings NV	Announcement Of 2017 Results And Update On The 2017 Audit Process	-111,46
5	23.10.2017	Stadio Holdings Ltd	Dealings in STADIO Shares by a Director of a Major Subsidiary	-125,12

Table 3.4: Extreme outlier events

What we can see with regards to these extreme outliers in Table 3.4 above is that events 1-4 are an example of clustered events of the infamous Steinhoff scandal. These extreme outliers, of which could provide informational value for supervised learning also distort the overall distribution, as observed in the statistical summary in Table 3.3, which can negatively affect the performance of the supervised learning model and hence will be omitted going forward.

3.3 Data Labelling

In order for us to proceed in building ML models, we will need to label each event. The events were labelled into one of the three classes, namely :

i CAR Significantly Positive CAR (Sig +)

ii CAR Significantly Negative CAR (Sig -)

iii Insignificant CAR (Insig)

The classifications were determined by using boxplots of the Event data omitting the extreme outliers. The three classes are defined as follows:

- Significantly Positive CAR are values greater than the third quartile by at least 1.5 times the InterQuartile Range (IQR)
- Significantly Negative CAR are values smaller than the first quartile by at least 1.5 times the IQR
- Everything else is then labelled as Insignificant CAR

Below is the distributions CAR as categorised by its respective classifications

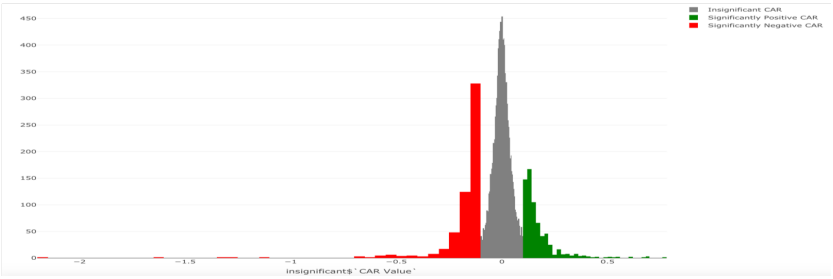

Figure 3.7: Distribution of CAR according to its classification

Below is a table summary of the respective classifications

Classifications	**Min**	**Max**	**Observations**	**% of Data**
Significantly Negative CAR	-219.20%	-10.70%	557	3%
Insignificant CAR	-10.60%	10.60%	18628	94%
Significantly Positive CAR	10.70%	76.13%	676	3%

Table 3.5: Data distribution per classification

What can be observed in Table 3.5 is that, as expected with outliers, the labelled data is imbalanced, meaning that most of the events are classified as Insignificant CAR. This nature of imbalanced data and how it will be handled will be discussed in Section 3.6.

3.4 Post-Labelling Discriminant Analysis

This Section, focuses on the first part of the overarching goal being :

to better understand the relationship between information dissemination and stock price trends in the SA markets

where we observe all the available features with the primary emphasis on the information dissemination feature (i.e *announcement type*) and try see whether there could be a relationship with the stock price movement, represented by the CAR classes.

3.4.1 Announcement Types

Below are the announcement types grouped by the 3 classes ranked from highest to lowest.

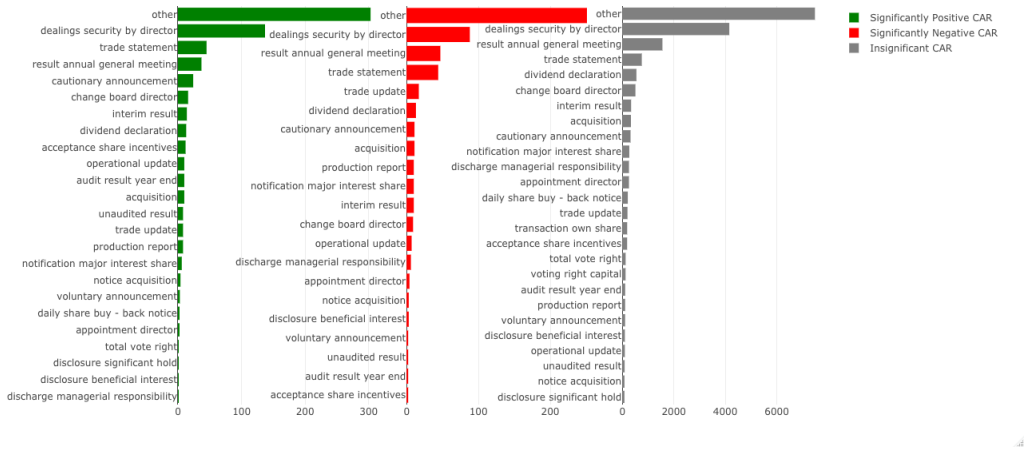

Figure 3.8: Announcement types of the various class Labels

What we can see from Figure 3.8 above is that the idiosyncratic announcement types dominate all the class types, this can potentially provide a challenge in classification performance as ideally we would like to see distinguishing announcement types in each of the classes. If we compare the significant CAR classes we see that although *dealings security by director* features as a top announcement type, it appears more often in the positive than in the

negative side of the CAR, which could indicate more often than not it would provide a positive CAR.

3.4.2 Sector

Below are the announcements tallied by Sector, for the 3 respective classes

Sectors	**Sig +**	**Sig -**	**Insig**	**Total**
Basic Materials	240	216	4306	**4762**
Communications	29	24	810	**863**
Consumer, Cyclical	107	69	2741	**2917**
Consumer, Non-cyclical	98	91	3794	**3983**
Diversified	3	4	359	**366**
Energy	27	16	265	**308**
Financial	79	63	4279	**4421**
Industrial	69	50	1571	**1690**
Technology	24	19	503	**546**
Total	**676**	**552**	**18628**	**N=19856**
χ^2				**232.04**
ϕ_c				**0.076**
p-value				**< 2.2e-16**

Table 3.6: Sector frequency table and association statistics

What we anticipate to see for an effective feature is some sectors reflecting high in exclusive classes, for example if a sector exclusively reflected highly in the Sig + class, it would imply that the sectors announcement more often than not, would reflect a positive announcement.
What we can see from Table 3.6 above is that *basic materials* is the most frequent sector across all three classes, which is due to the high amount of *basic materials* announcements. All 3 classes generally have the same sectors (*financial, industrial, consumers*) which implies that there is no distinguishing sector in any of the 3 classes, which is evident with the low Cramér's V (ϕ_c) value of 0.076, making it an ineffective feature.

3.4.3 Market Capitalisation

Below is the distribution of market capitalisation of the events respective companies as at 29 July 2019 grouped by the 3 classes

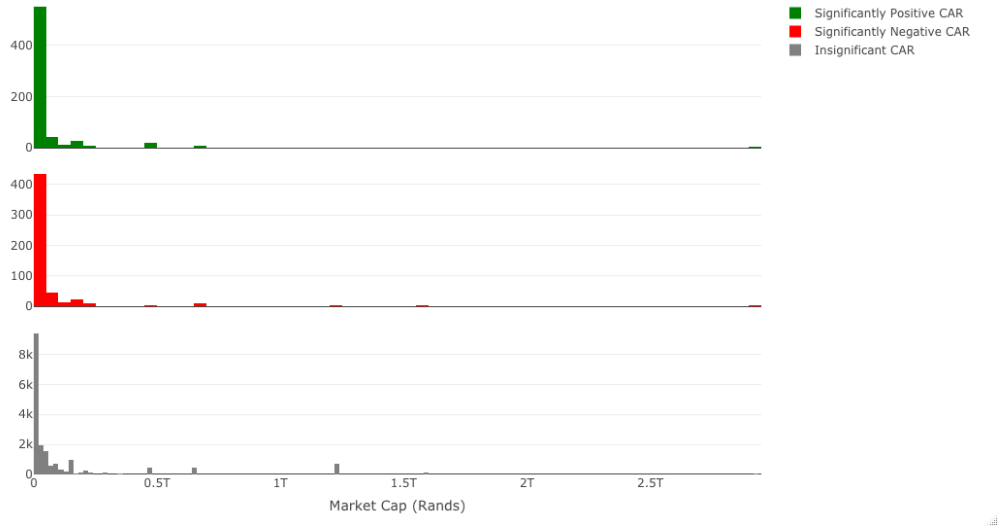

Figure 3.9: Market capitalisation of the various class labels

What we can see from Figure 3.9 above is that most of the announcements across the 3 classes occur in the lower range of the spectrum of liquid companies with a market cap ranging from R 1 billion to R 100 billion. On the extreme end of R2.9 trillion there is representation from all 3 classes although magnitudes less than the lower end. Implying that there could be no heterogeneity across the classes.

3.5 Dataset

The data is subdivided as follows:

- 60% of the data is allocated for training. In absolute terms that is 11880 announcements.
- 20% of the data is allocated for evaluation. In absolute terms that is 3988 announcements.
- 20% of the data is allocated for testing. In absolute terms that is 3988 announcements.

Below is a visualisation of the training data broken down by the 3 classes

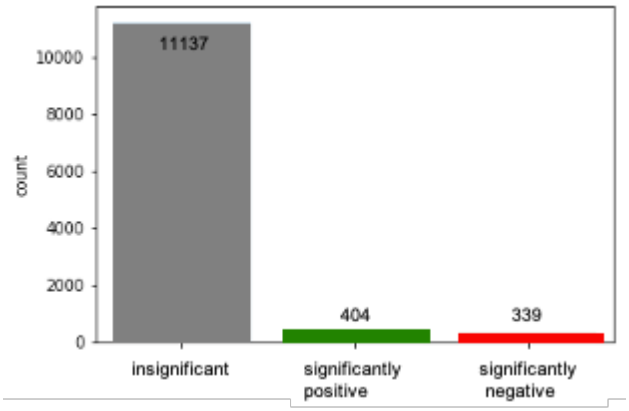

Figure 3.10: Training data distribution of classes

As viewed in Figure 3.10 above, there is a clear sign of imbalanced data as alluded to in Section 3.3. We will be investigating a few strategies to cater to the imbalanced data in the next Section.

3.6 Imbalanced Data

The issue with imbalanced data is that with supervised learning, the trained models will typically provide a high training classification accuracy which is optically deceiving as the model is overfitting the training data of which can be verified through the test set.

There are numerous methods of dealing with imbalanced data (Sun, A. K. Wong, and Kamel 2009). The approaches investigated while training the models were the following:

- **Downsampling** - this method involves removing random events from the majority class (Insig), the downside of this method is that it can cause loss of information. Figure 3.11 below illustrates the application of this method with respect to the training data.

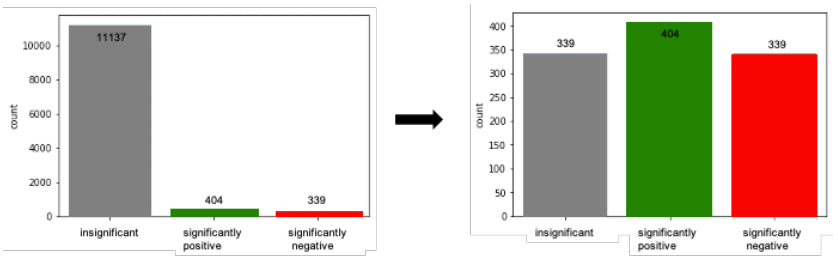

Figure 3.11: Downsampling distribution of classes

- **Upsampling** - this method involves duplicating random events from the minority classes (Sig +, Sig -), the downside of this method is that it can cause overfitting. Figure 3.12 below illustrates the application of this method with respect to the training data.

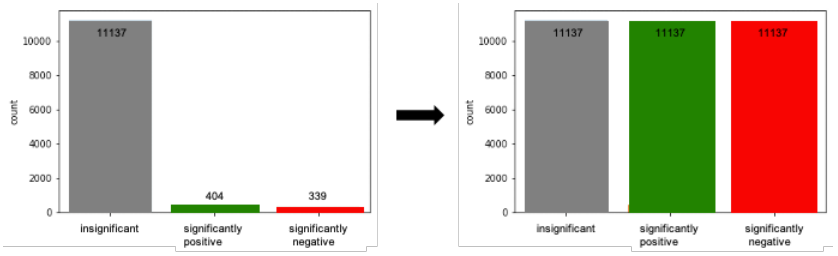

Figure 3.12: Upsampling distribution of classes

- **Cost-Sensitive Training** - this involves using an asymmetric cost function to artificially balance the training process. In the case of this method, the training data remains as is.

The above-mentioned methods to adress class imbalance are implemented in Section 6.1 in the form of an experiment on a subset of the data to observe how the models fair and the best performing technique is then applied across the full dataset.

Chapter 4

Machine Learning Models

4.1 Model Building

Although a multitude of models have been used in the literature discussed earlier, only a subset was implemented based on the availability of pre-built estimators in the Tensorflow eco-system in order to leverage TensorFlows' USE for the text features.

The available models capable of multi-class classification that were implemented were the following :

- A classical machine learning model - Logistic Regression (LR)
- A deep learning model - Deep Neural Network (DNN)

4.1.1 Logistic Regression

This method serves as the foundation of supervised learning methods and hence it is important to include into model comparisons (Hastie, Tibshirani, and Friedman 2009). As there are many ways to derive this model, in particular, we will be looking at the Multinomial Logistic Regression (MLR) as we have 3 classes (Sig +, Insig, Sig -). The target classes are mapped as

follows: $\mathbf{t} \in \{1 : \text{Sig +}, 2 : \text{Insig}, 3 : \text{Sig -}\}$. We will emphasise it in matrix form as this is how it is implemented through the use of computational graphs, as illustrated in Figure 4.2, which is the backbone behind Tensorflow.

Assuming $\mathbf{x}$ is the input feature vector of length $\mathbf{M}$ and $\mathbf{t}$ is the target label represented as one hot vector equal to 1 for the correct class $\mathbf{c}$ and 0 everywhere else (i.e $t = \begin{bmatrix}0 & 1 & 0\end{bmatrix}$).

We can apply a linear model to $\mathbf{x}$ to obtain the $\mathbf{z}$ vector which represents the scores of the 3 classes.

$$\mathbf{z} = W\mathbf{x} + \mathbf{b} \tag{4.1}$$

where $\mathbf{W}$ is a $3 \times \mathbf{M}$ matrix representing the weights (coefficients) and $\mathbf{b}$ is a length 3 vector of biases.

We then convert these scores into probabilities that sum to 1 using the softmax function σ which takes as an input the 3-dimensional vector $\mathbf{z}$ and outputs a 3-dimensional vector $\hat{\mathbf{y}}$ of real values between 0 and 1 which represents the predicted class distribution.

$$\hat{y_c} = \sigma(\mathbf{z})_c = \frac{e^{z_c}}{\sum_{d=1}^{3} e^{z_d}} \quad \text{for } c = 1 \cdots 3 \tag{4.2}$$

We then compare the target $\mathbf{t}$ to the predicted $\hat{y}$ to determine the loss and subsequently make adjustments to the weights $\mathbf{W}$ and $\mathbf{b}$ iteratively via gradient descent to decrease the eventual loss over a series of training steps. The loss function used is the cross-entropy loss function ξ. How we get to this loss function is from starting at the likelihood function.
Given a set of parameters θ of the model that can result in the prediction of the correct class for each input sample. The maximisation of this likelihood can be written as follows:

$$\underset{\theta}{\text{argmax}}\ \mathcal{L}(\theta|\mathbf{t}, \mathbf{z}) \tag{4.3}$$

The likelihood $\mathcal{L}(\theta|\mathbf{t}, \mathbf{z})$ can be rewritten as the joint probability of generating $\mathbf{t}$ and $\mathbf{z}$ given the parameters θ: $P(\mathbf{t}, \mathbf{z}|\theta)$. This can be written as a conditional distribution:

$$P(\mathbf{t}, \mathbf{z}|\theta) = P(\mathbf{t}|\mathbf{z}, \theta)P(\mathbf{z}|\theta) \tag{4.4}$$

Since we are not interested in the probability of $\mathbf{z}$, we can therefore reduce this to:

$$\mathcal{L}(\theta|\mathbf{t},\mathbf{z}) = P(\mathbf{t}|\mathbf{z},\theta) \tag{4.5}$$

This can be written as $P(\mathbf{t}|\mathbf{z})$ for fixed θ. Since each t_c is dependent on the full $\mathbf{z}$ and only 1 class can be activated in the $\mathbf{t}$, we can therefore write:

$$P(\mathbf{t}|\mathbf{z}) = \prod_{i=c}^{3} P(t_c|\mathbf{z})^{t_c} = \prod_{c=1}^{3} \sigma(\mathbf{z})_c^{t_c} = \prod_{c=1}^{3} \hat{y_c}^{t_c} \tag{4.6}$$

Maximising the likelihood is equivalent to minimising the negative log likelihood which brings us to the following:

$$\mathcal{L}(\theta|\mathbf{t},\mathbf{z}) = -\log \mathcal{L}(\theta|\mathbf{t},\mathbf{z}) = \xi(\mathbf{t},\mathbf{z}) = -\log \prod_{c=1}^{3} \hat{y_c}^{t_c} = -\sum_{c=1}^{3} t_c \cdot \log(\hat{y_c}) = L_i \tag{4.7}$$

We augment the loss function derived in Equation 4.7 with a regularization term that tries to trade off over-fitting the training data and preferring simpler models. This is done by tuning the regularisation strength λ hyperparameter.

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + \lambda R(W) \tag{4.8}$$

There are a few regularisation methods, the most common is the L2 regularisation whereby one is penalising the Euclidean norm of the weight vector $\mathbf{W}$.

$$R(W) = \sum_{k=1}^{3} \sum_{l=1}^{M} W_{k,l}^2 \tag{4.9}$$

The second regularization method is the L1 regularization whereby one is penalising the L1 norm of the weight vector $\mathbf{W}$. The L1 regularization has a nice property in that it encourages sparsity in $\mathbf{W}$ which means eliminating features that are not contributing to the improvement of the model, of which the L2 regularization struggles to do.

$$R(W) = \sum_{k=1}^{3} \sum_{l=1}^{M} \mid W_{k,l} \mid \tag{4.10}$$

Both these regularisation methods discussed above are used in this thesis and the implementation are discussed in Section 6.1.

Figure 4.1 below is a simplistic illustration of the MLR computational graph as well as a cross entropy loss calculation as visual reference to the concepts discussed in this section.

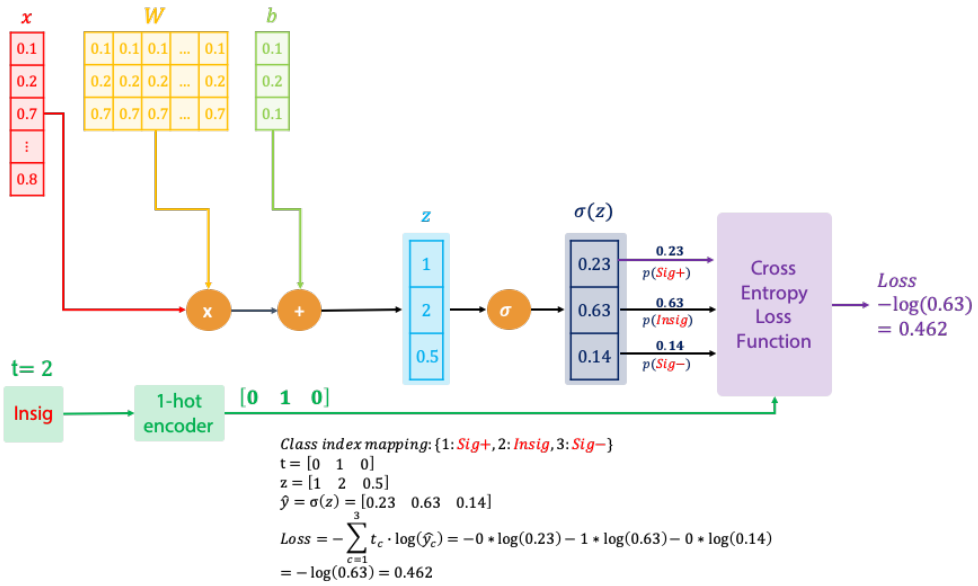

Figure 4.1: An example of a LR computational graph and cross entropy loss calculation

The simplicity of LR makes it an ideal candidate as a benchmark model to measure performance, however with higher dimensional data-sets that exhibit non linear properties, it tends to under-perform due to its linear decision surface. This range of problems requires a generalized form of the LR which will be discussed in the next section.

4.1.2 Deep Neural Network

In order to discuss DNN, we first need to discuss it in its fundamental form, starting as an Artificial Neural Network (ANN).

ANNs are computational models that imitate the way biological neural networks process information in the brain. At its most basic form, it consists of multiple neurons. A neuron j receives numerous input signals $x_1, x_2, ..., x_m$ from the other neurons, each of which are multiplied by a weight w_{ik} and summed. A bias weight b_j is also added to provide an additional parameter to improve model fit. The output z of a single neuron j is defined as follows:

$$z_j = \sum_{k=1}^{m} w_{jk} x_j + b_j \tag{4.11}$$

The final output signal of neuron j is computed by feeding z_j through an activation function $g(z)$. The simplest type of neuron, called a *perceptron*, uses a binary activation function called the Heavyside step function, which outputs 1 if input is positive and 0 if it is negative.

A single perceptron is able to approximate linear functions. By constructing a network of perceptrons, like the LR model discussed in Section 4.1.1 which is in fact a single layer perceptron, one can approximate more relevent non-linear functions. Taking it a step further, by constructing one or more of these network of perceptrons in one or more hidden layers using a non-linear activation function, we get a MLP also known as a Feed Forward Neural Network or Neural Network (NN) for short. The distinction between NN and a DNN is that a DNN has more than one hidden layer. An illustration of this can be viewed in Figure 4.2 below.

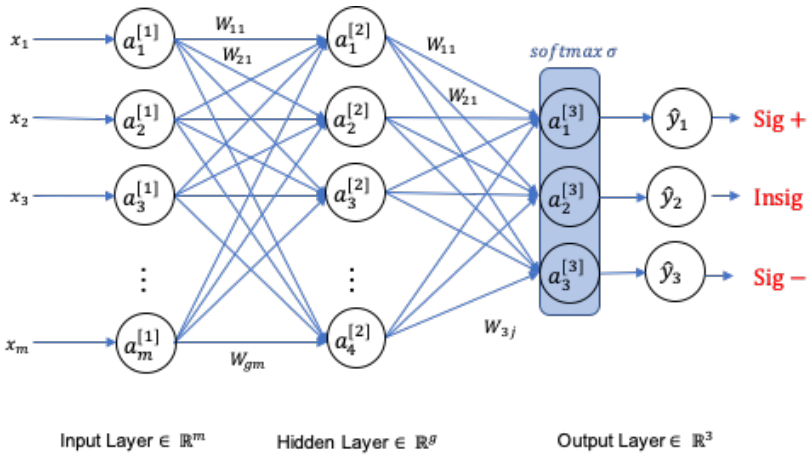

Figure 4.2: Diagram of a MLP with m input nodes, j hidden nodes and one hidden layer with 3 output nodes, the output activation function is a softmax, as with the LR discussed in 4.1.1. For better aesthetic, the bias node was not shown.

Looking at a single neuron, as with LR, the computation of z (called the pre activation value) for the j^{th} neuron on the l^{th} layer is

$$z_j^{[l]} = \sum_k w_{jk}^{[l]} a_k^{[l-1]} + b_j^{[l]} \tag{4.12}$$

The output of neuron (activation value $a_j^{[l]}$) j in the l^{th} layer is computed by feeding the pre-activation value $z_j^{[l]}$ through an activation function $g(z)$. The activation function used in this project is the Rectified Linear Unit (ReLU), which replaces negative z values with 0.

$$a_j^{[l]} = g^{[l]}(z_j^{[l]}) \tag{4.13}$$

where $g^{[l]}$ denotes the l^{th} layer activation function. Similarly with the LR as defined in Equation 4.2, the last hidden layer is passed through a softmax function σ to produce a 3-dimensional vector $\hat{\mathbf{y}}$ of real values between 0 and 1 which represents the predicted class distribution.

$$\hat{y} = \sigma(W^{[L]} A^{[L-1]} + b_2) \tag{4.14}$$

where $L-1$ is the last hidden layer in the network.

As with the LR discussed in Section 4.1.1, the resultant $\hat{y}$ is compared to the target $\mathbf{t}$ via the loss function and in particular the cross-entropy loss function (Equation 4.7). The entire computation process through the layers that has been discussed is called the *forward-propagation*.

Next, the summed divergence in the output layer is propagated back through the network in order to calculate the gradients by using the *back-propagation* algorithm. These gradients are used by an optimization technique, like the *Adam* used in this thesis, in order to minimise the loss function by adjusting the weights $\mathbf{w}$ across the entire network. To obtain the function minimum, most optimization techniques at its core utilise *gradient descent*, which works by taking steps proportional to the negative direction of the gradient at the current point. When all the training examples have been processed in this manner, the network has completed one *epoch*. This procedure can be repeated for numerous epochs to further tweak the weights.

4.2 Hyper-parameter Tuning

When working with machine learning models, determining which hyper-parameters to use is a non-trivial task. The process of determining the optimal hyper-parameters can be automated through a process called hyper-parameter tuning. A hyper-parameter tuning algorithm searches the hyper-parameter space for a combination of hyper-parameters that optimizes some objective function, typically a metric like accuracy on a evaluation set for classification problems. The models are trained on a training set with some combination of hyper-parameters for a number of epochs, before being evaluated on an evaluation set, the process is then repeated with a different combination of hyper-parameters. The combination that yields the best performances on the evaluation set is returned. There are numerous tuning algorithms, the algorithm used in this thesis, which has shown to outperform all other methods (Bergstra and Bengio 2012), is the Bayesian Optimization (BO).

Bayesian Optimization searches the hyper-parameter space intelligently by modelling distributions over objective functions as Gaussian Processes (GP). Given the model performance on an evaluation set as a function of hyper-parameters, $f(x)$, the task is to optimize $f(x)$ by selecting the best hyper-

parameters, this is shown in Equation 4.15 below :

$$x^* = \arg\max_{x \in X} f(x) \tag{4.15}$$

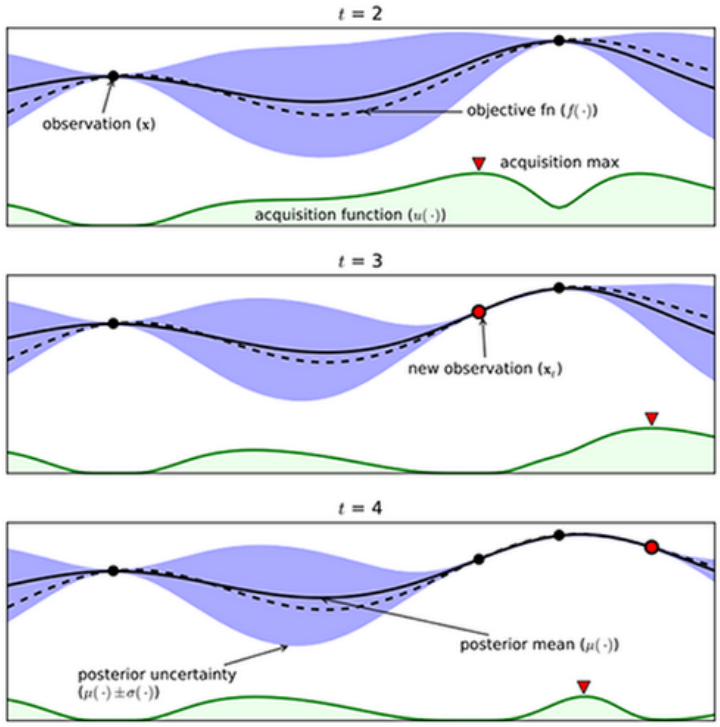

Figure 4.3: Gaussian process approximation of objective function (Brochu, Cora, and De Freitas 2010)

Looking at Figure 4.3 above, the dotted line represents an unknown objective function of which the BO tries to find the max. The BO selects some point on the graph where the mean (exploitation) and the variance (exploration) is high, this creates a exploitation-exploration trade-off that is encoded as an acquisition function that provides a single measure of how useful it would be to try any given point. This acquisition function is shown as the green surface in the figure. The acquisition function used on Google's Cloud ML is the Expected Improvement (EI) function, which measures the expected increase in the objective function given the next point pick. EI is defined as $EI(x) = E[max\{0, f(x) - f(\hat{x})\}]$ where $\hat{x}$ is the current best combination of

hyper-parameters. In summary, the following steps are run for N iterations when using BO:

1. Update posterior expectation of f given observed values $f(x)$ using Gaussian Process (GP).
2. Find x_i which optimizes EI: $x_i = argmax EI(x)$.
3. Compute value of f for x_i.

The above-mentioned algorithm was used through Google Cloud ML's HyperparameterTuning module on three scenarios to cater for the imbalanced data discussed in Section 3.6.

Chapter 5

Implementation

In this section, we introduce vital implementation details of this thesis. The implementation consists of numerous processes which are Data Pre-processing, Model Building, Model Operation and Model Evaluation. The implementation architecture is illustrated in Figure 5.1.

5.1 Input Data

The data acquisition of Text and Market data is discussed in detail in Section 3.1.1 and 3.1.2 respectively.

5.2 Data Pre-processing

This process consists of four main data pre-processing sub-processes and the final data set ready for model building. These sub-processes are: Market Data Liquidity Filter, Categorical Feature Extraction, Announcements (Text)-Market Mapping, Event Study (y Labelling) and the Final Dataset.

5.2.1 Market Data Liquidity Filter

The first process after acquiring both the text and market data is to filter out illiquid companies because in a trading environment, where this model would

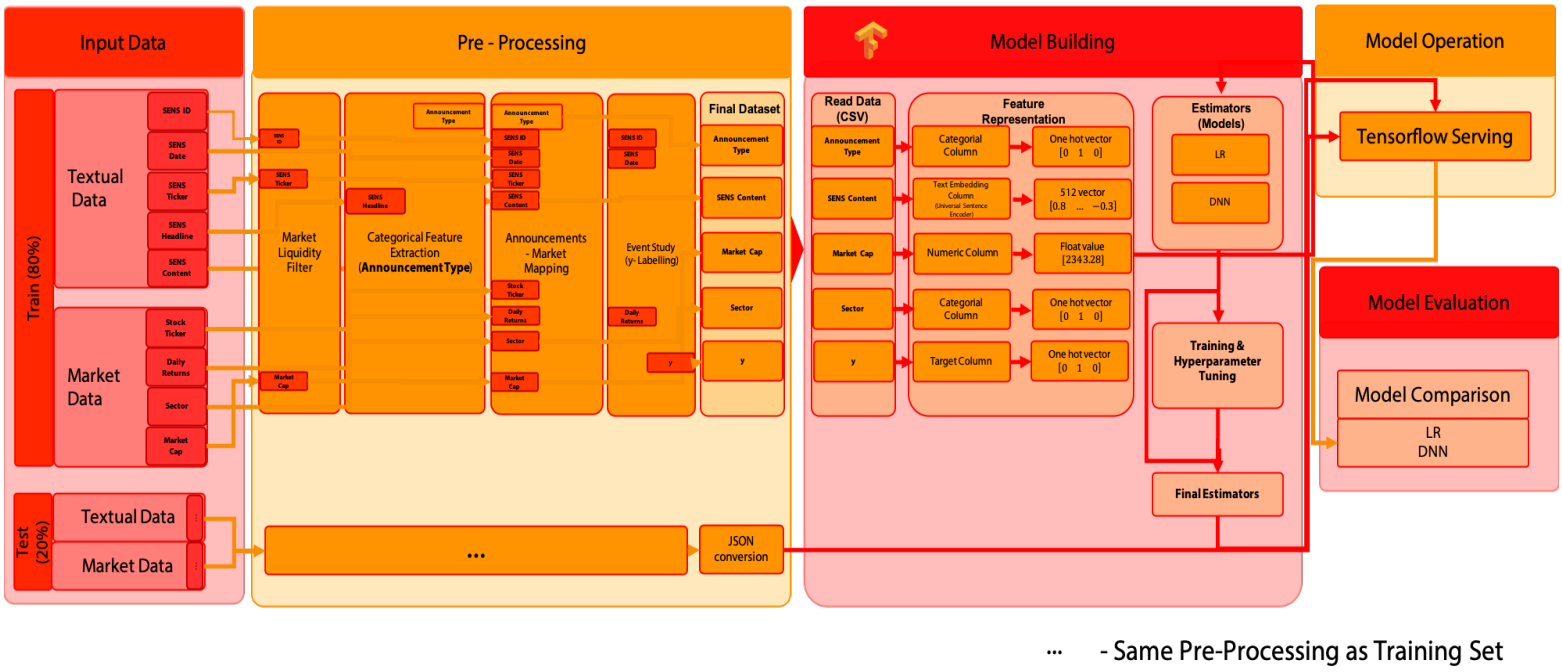

Figure 5.1: Overview of overall implementation architecture

be implemented, we would not be able to enter/exit our position (buy/sell) fast enough to capitalise on the abnormal returns opportunity. The filtration process has been discussed in detail in Section 3.1.1.

5.2.2 Categorical Feature Extraction

As discussed in Sections 3.1.1 and 3.1.4, there is a need for some categorical features derived from the text to potentially assist the ML models in improving its classification performance, as a text feature (*sens_content*) alone could potentially be cumbersome. Due to the lack of standardisation of *announcement types*, lemmatization is applied to cluster *sens_headline* and the top 24 *announcement types*, which represents 59% of the *announcement types*, are kept as is and the rest were labelled as *other*.

5.2.3 Announcements (Text)-Market Mapping

In this phase, we primarily link the textual data features (*sens_content, announcement_type*) with the market data features (*market_cap, sector*).

5.2.4 Event Study (y Labelling)

As discussed in detail in Section 3.2, for each announcement, a window of daily returns surrounding the announcement is used to label the announcement, by applying the event study methodology, into one of the three classes, being: Sig +, Insig, Sig -.

5.2.5 Final Dataset

After all the above mentioned processes are conducted, we finally have a data set ready for model building and evaluation. The final data sets' details are tabulated in Table 5.1 below.

Feature	Data Type	Description	Example
announcement_type	categorical	The announcement type derived from the *sens_headline* field.	trade statement
sens_content	string	The contents of the SENS announcement	RFG 201411130005A Trading Statement...
market_cap	numerical	Market Capitalisation (as at 15 July 2019) of the particular company related to the announcement.	316152097.85
sector	categorical	The sector of the particular company related to the announcement as per the Industry Classification Benchmark (ICB).	Financial
y	categorical	The announcement label.	Insig

Table 5.1: Descriptions of features in the final dataset

For model building, the data set is converted to Comma-Separated Values (CSV) format for model ingestion on Tensorflow.

5.3 Model Building

5.3.1 Feature Representation

Upon data ingestion, the features are represented in various ways depending on the data type and the feature engineering strategies applied. Below we will be discussing how the features are represented for the models to ingest and how the output label will be represented.

- ***sens_content*** - as detailed in Section 3.1.1, this textual feature will be represented as a standard 512 vector via the USE.
- ***announcement_type*** - given that the vocabulary size is small (25), the feature is therefore represented as a one hot encoded vector.
- ***sector*** - given that the vocabulary size is small (10), the feature is therefore represented as a one hot encoded vector.
- ***market_cap*** - this feature is represented as a numerical value. Normalisation was not applied to this vector.
- ***y*** - given that there are only 3 outcomes, the output is represented as a one hot encoded vector.

5.3.2 Estimators (Models)

The *Estimators*[1] are a high-level TensorFlow API which encapsulates training, evaluation and serving of models. Combining Estimators with a scalable computation of Google Cloud's AI-Platform allows one to train, evaluate and serve models that would otherwise take a significant amount of time to implement on a local PC. There are two Estimators used in this thesis, primarily due to the limitation of Estimators compatible for multiclass classification. The Estimators used, as discussed in detail in Section 4.1, are the Logistic Regression (LR) and the Deep Neural Network (DNN).

[1] https://www.tensorflow.org/guide/estimator

5.3.3 Training & Hyper-parameter Tuning

Training of the above mentioned models were conducted in a phased approach.
The first phase was to cater for the dataset imbalance and to ensure that the models are adequately trained with that in mind. This was conducted through a set of experiments looking at the three model imbalance training methods, elaborated in Section 3.6, and comparing the best models through the process of hyper-parameter tuning on a subset of the dataset, the results of the experiment are detailed in Section 6.1. In the second phase we select the best models as well as the best training method and train across the whole dataset for final comparison.

5.3.4 Final Estimators

As discussed in the previous subsection, once the Hyper-parameter tuning is completed with the ideal training method, the final models are then trained across the entire dataset and once this is complete, then the models are ready for operation.

5.4 Model Operation

The trained models are operationalised through the process of packaging and hosting on Google Cloud AI Platform, where it can be accessed at scale via an API call and parsing the test data in JavaScript Object Notation (JSON) format and returning the prediction in seconds.
The test set, as discussed in Section 5.5.2, is parsed into the LR and DNN operationalised models and the predictions are returned for final evaluation.

5.5 Model Evaluation

Upon the completion of model training, we need to evaluate the performance of these models by a set of metrics and by using these metrics on out of sample data, we will be discussing these two in the upcoming subsections.

5.5.1 Evaluation Metrics

Standardised measures are required in order to evaluate the classification results on different learning algorithms used. In this section, we will discuss the several evaluation approaches used to observe the performance of the classification methods.

The Confusion Matrix

Predictions made by a classifier can be expressed in the form of a confusion matrix, where each entry contains the number of correct/incorrect predictions. These entry types are defined in the table below :

Entry	Definition
$TrueSig+$	Correctly classified Significantly Positive CAR (Sig +)
$FalseSig+_{Insig}$	Incorrectly classified Significantly Positive CAR (Sig +). The actual class being Insignificant CAR (Insig)
$FalseSig+_{Sig-}$	Incorrectly classified Significantly Positive CAR (Sig +). The actual class being Significantly Negative CAR (Sig -)
$FalseInsig_{Sig+}$	Incorrectly classified Insignificant CAR (Insig). The actual class being Significantly Positive CAR (Sig +)
$TrueInsig$	Correctly classified Insignificant CAR (Insig)
$FalseInsig-_{Sig-}$	Incorrectly classified Insignificant CAR (Insig). The actual class being Significantly Negative CAR (Sig -)
$FalseSig-_{Sig+}$	Incorrectly classified Significantly Negative CAR (Sig -). The actual class being Significantly Positive CAR (Sig +)
$FalseSig-_{Insig}$	Incorrectly classified Significantly Negative CAR (Sig -). The actual class being Insignificant CAR (Insig)
$TrueSig-$	Correctly classified Significantly Negative CAR (Sig -)

Table 5.2: Confusion Matrix entry definitions

The Table 5.3 below corresponds to a 3 x 3 confusion matrix : the columns represent the predicted class and the rows represent the actual class. The diagonal entries contain the total number of correct predictions in each class, whereas the remaining entries represent the number of misclassifications as elaborated in Table 5.2.

		Predicted		
		Sig +	**Insig**	**Sig -**
Actual	**Sig +**	$TrueSig+$	$FalseInsig_{Sig+}$	$FalseSig-_{Sig+}$
	Insig	$FalseSig+_{Insig}$	$TrueInsig$	$FalseSig-_{Insig}$
	Sig -	$FalseSig+_{Sig-}$	$FalseInsig_{Sig-}$	$TrueSig-$

Table 5.3: Multi-Class Confusion Matrix

Accuracy

The accuracy metric in the multiclass context is the sum of the diagonals divided by the sum of all the entries in the confusion matrix which is illustrated in Equation 5.1 below:

$$Accuracy = \frac{TrueSig+ + TrueInsig + TrueSig-}{TotalSampleSize} \tag{5.1}$$

The problem with the above metric in the context of an imbalanced dataset is that a significant $TrueInsig$ will provide optically high accuracy due to it being the majority class (Insig). What this means in model training is that the model will bias classifying the Insig class, implying in the model application that no trading decision for all events. We are concerned in correctly classifying the minority classes (Sig +, Sig -) as they present trading opportunities to generate abnormal returns. Precision and recall and the combination of both can cater to the minority classes.

Precision

Precision looks at what proportion of a predicted class is predicted correctly. For example the Sig + class: we look at the correctly classified Sig + ($TrueSig+$) over all the predicted Sig + class where it is correct or not ($TrueSig+, FalseSig+_{Insig}, FalseSig+_{Sig-}$). This is illustrated in Equation 5.2 below:

$$Precision_{Sig+} = \frac{TrueSig+}{TrueSig+ + FalseSig+_{Insig} + FalseSig+_{Sig-}} \tag{5.2}$$

Similarly, we can calculate the precision of the other two classes which is

illustrated in Equations 5.3 and 5.4 below :

$$Precision_{Insig} = \frac{TrueInsig}{FalseInsig_{Sig+} + TrueInsig + FalseInsig_{Sig-}} \tag{5.3}$$

$$Precision_{Sig-} = \frac{TrueSig-}{TrueSig++FalseSig+_{Insig}+FalseSig+_{Sig-}} \tag{5.4}$$

Recall

Recall looks at what proportion of an actual class is predicted correctly. For example the Sig + class : we look at the correctly classified Sig + ($TrueSig+$) over all the actual Sig + class ($TrueSig+, FalseInsig_{Sig+}, FalseSig_{Sig+}$). This is illustrated in Equation 5.5 below:

$$Recall_{Sig+} = \frac{TrueSig+}{TrueSig++FalseInsig_{Sig+}+FalseSig-_{Sig+}} \tag{5.5}$$

Similarly, we can calculate the recall of the other two classes which is illustrated in Equations 5.6 and 5.7 below :

$$Recall_{Insig} = \frac{TrueInsig}{FalseSig+_{Insig}+TrueInsig+FalseSig-_{Insig}} \tag{5.6}$$

$$Recall_{Sig-} = \frac{TrueSig-}{TrueSig--+FalseSig+_{Sig-}+FalseInsig_{Sig-}} \tag{5.7}$$

F1-Score

Ideally, we prefer to have models with high precision and recall scores. However there is a trade-off, when we increase the recall, we decrease the precision and visa-versa. We typically want to incorporate both metrics in an optimal blend which we can use what is called the F1 score.
The F1 score is the harmonic mean of precision and recall which is defined on a per class level by the following Equations (5.8, 5.9, 5.10) below :

$$F1_{Sig+} = \frac{2 \times Precision_{Sig+} \times Recall_{Sig+}}{Precision_{Sig+} + Recall_{Sig+}} \tag{5.8}$$

$$F1_{Insig} = \frac{2 \times Precision_{Insig} \times Recall_{Insig}}{Precision_{Insig} + Recall_{Sig+}} \tag{5.9}$$

$$F1_{Sig-} = \frac{2 \times Precision_{Sig-} \times Recall_{Sig-}}{Precision_{Sig-} + Recall_{Sig-}} \tag{5.10}$$

Utilising the harmonic mean ensures that we penalise extreme precision and recall values. We can combine the per-class F1-scores into a single number, which represents the model's overall F1-score. The 3 ways are as follows :

1. Micro-averaged F1-score (Micro avg)
2. Weighted-average F1-score (Weighted avg)
3. Macro-averaged F1-score (Macro avg)

Micro-averaged F1-score

In order to compute the micro-averaged F1-score we first need to calculate the micro-averaged precision and recall which is defined as follows :

$$Micro\ Precison = \frac{\sum_c True\ Positive_c}{\sum_c True\ Positive_c + \sum_c False\ Positive_c} \tag{5.11}$$

$$Micro\ Recall = \frac{\sum_c True\ Positive_c}{\sum_c True\ Positive_c + \sum_c False\ Negative_c} \tag{5.12}$$

where **c** is the class label consisting of {Sig +, Insig, Sig -}.

In a multi-class scenario, all the false instances are counted, this therefore implies that :

$$\sum_c False\ Positive_c = \sum_c False\ Negative_c \tag{5.13}$$

$$\therefore \quad MicroPrecision = MicroRecall = MacroF1 = Accuracy \tag{5.14}$$

Which brings us back to the discussion on accuracy in Section 5.5.1 and therefore will not be an ideal metric to use given our imbalanced dataset.

Weighted-average F1-score

As the name alludes to, this metric weights the F1-score of each class by the number of samples from that class. This is illustrated in Equation 5.15 below :

$$Weighted\, F1 = w_{Sig+}\, F1_{Sig+} + w_{Insig}\, F1_{Insig} + w_{Sig-}\, F1_{Sig-} \tag{5.15}$$

The above metric would not be ideal as more emphasis will be placed on the dominant class (Insig).

Macro-averaged F1-score

This score is the arithmetic mean of the F1 scores of the 3 classes, this is computed irrespective of their individual weightings.

$$Macro\, F1 = \frac{F1_{Sig+} + F1_{Insig} + F1_{Sig-}}{3} \tag{5.16}$$

5.5.2 Test Set

The out of sample test set represents the 20% of data reserved from the total available data. The illustration of the label breakdown can be viewed in Figure 5.2 below.
What we can see in Figure 5.2 below is a true reflection of the total data distribution, being imbalanced, as well as how the typical financial markets behave; a series of normal events followed by abnormalities that our models aspire to detect and capitalise on. The test set is converted to JSON format as viewed in Listing 1 below and parsed into TensorFlow serving model, as discussed in Section 5.4, for prediction.

```
{"sens_content": "RFG 201411130005A Trading Statement...",
"announcement_type": "trade statement",
"sector": "Financial",
"market_cap": 31611052097.85}
```

Listing 1: A test set example in JSON format

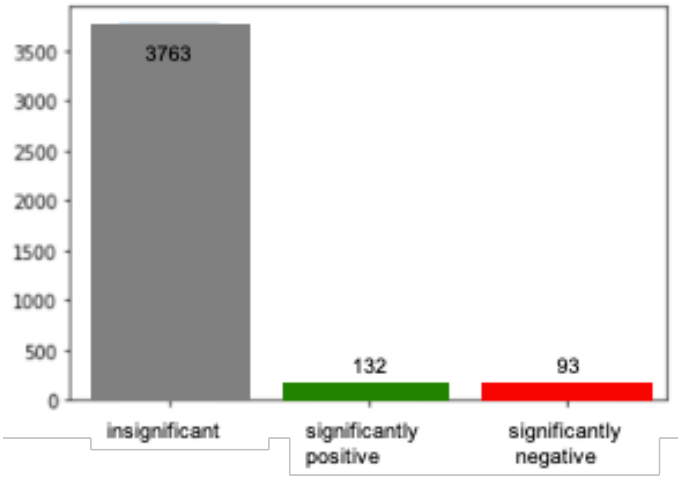

Figure 5.2: Test set labels

5.6 Frameworks, libraries and programming languages

The Textual and Market data extraction and initial wrangling was done in the *R* programming language. The libraries and implementation is detailed in Section 3.1.2 and Section 3.1.1.
The developments of the ML models were done using the *Python* programming language version 2. This language has become the standard when working with ML as most ML libraries are developed primarily for this language. In addition, its readable programming language works interoperably with low level C code, making it efficient for complex mathematical operations.
For visualisations, the *Plotly*[2] library was primarily used on both *Python* and *R* environments.
Further data wrangling for model building and analysis was conducted using the *Pandas*[3] library.
The ML models were implemented with *TensorFlow*[4], which has become the most popular framework for ML. Developed by Google, it features mul-

[2] https://plotly.com/
[3] https://pandas.pydata.org/
[4] https://www.tensorflow.org/

tiple levels of abstraction for implementing a vast range of ML models. The pre-built Estimators were used, which are fully-equipped ML models following best practices that require minimal coding, as opposed to the low-level libraries which provide endless options for customisation.
The models were trained in the cloud, where large clusters of GPU-equipped computers can train and optimise models at a fraction of the time it would have taken on a regular desktop, this is done via Google $AI - Platform$[5].

[5] https://cloud.google.com/ai-platform

Chapter 6

Experiment Results and Analysis

This chapter focuses on the second part of the overarching goal being :

To utilise the relationship, in goal 1, to forecast price changes with the eventual goal of assisting market participants in their trading decisions

We go about achieving this goal through a series of experiments.

1. **Experiment 1** - the outcome of this experiment fills two needs with one deed; firstly, by addressing data imbalance by looking at 3 standard methods framed as scenarios. Secondly, finding the best model parameters for the scenarios via hyper-parameter tuning.

2. **Experiment 2** builds on the learning's of **Experiment 1** by selecting the best model parameters of the preferred scenario and training those models on the entire dataset and finally evaluated on the out of sample test set.

3. **Experiment 3** builds on the learning's of **Experiment 2** by looking at what attributes could possibly affect the performances of these models. This is exectued by a series of hyperparameter tuning jobs followed by

training the optimal models on the entire dataset and finally evaluated on the out of sample test set.

6.1 Experiment 1 : Hyper-parameter Tuning taking into account Class Imbalance

The first experiment consists of three scenarios which are the three commonly used methods to address class imbalance, these methods have been discussed in detail in Section 3.6. They are the following :

1. **Scenario 1** : Model training and evaluating using Upsampled data.
2. **Scenario 2** : Model training and evaluating using Downsampled data.
3. **Scenario 3** : Model training and evaluating using data as is and implementing a cost function to penalise the overweight class.

The model configurations and hyper-parameters for these three scenarios are summarized in Table 6.1 below :

Parameter	Description	Logistic Regression	DNN
batch-size	Batch size for each training and evaluation step	[5,...,90]	[20,...,100]
learning-rate	Learning rate value for the optimizers	[0.01,...,0.1]	[0.015,...,0.08]
layer-sizes-scale-factor	Determine how the size of the layers in the DNN decays	-	[0.3,...,1]
num-layers	Number of layers in the DNN	-	[3-6]
hidden-units	Size of the first Hidden Layer	-	[150,...,800]
l1-strength	L1 Regularization Strength	[0.01,...,0.9]	-
l2-strength	L2 Regularization Strength	[0.01,...,0.9]	-

Table 6.1: Hyper-parameters for DNN and LR models.

Hyper-parameter tuning works by running multiple trials in a single training job. Each trial is a complete execution of your training application with values for your chosen hyper-parameters, set within limits specified, as viewed in Table 6.1 above. The AI Platform Training training service keeps track of the results of each trial and makes adjustments for subsequent trials. When the job is finished, a summary of all the trials along with the most effective configuration of values according to the criteria initially specified as well as a visual representation of the trials via Learning Curves (LC) are observed to compare the performance behaviour over the course of the model training.

Scenario 1 : Upsampling

Below are the results of the hyper-parameter training of 5000 training samples on 3 epochs.

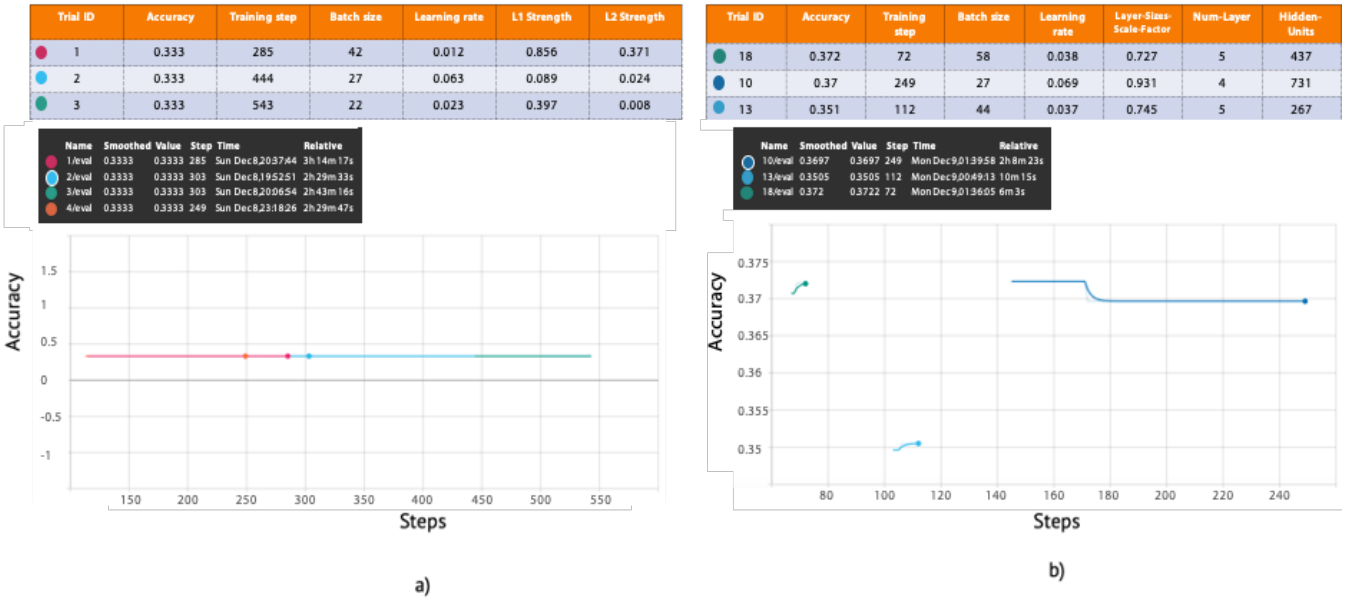

Figure 6.1: Scenario 1 : Top models and their parameters based on evaluation accuracy as a function of the training steps a) LR b) DNN

If we observe the top performing models out of a total of 20 trials of the LR in Figure 6.1a above, we can see that the models all plateau early on in their training at an evaluation accuracy of 0.33 or 33% which is typically chance results given that there are 3 classes, which makes it as as good as a three sided dice.
If we observe the top performing models out of a total of 20 trials for the DNN in Figure 6.1b above, they marginally outperform the LR and the evaluation learning curve generally plateaued with the various step durations given the different batch sizes.
Overall, we can see that upsampling our data provides us with subpar model performances

Scenario 2 : Downsampling

Below are the results of the hyper-parameter training of 5000 training samples on 3 epochs.

Figure 6.2: Scenario 2 : Top models and their parameters based on evaluation accuracy as a function of the training steps a) LR b) DNN

If we observe the top performing models out of a total of 20 trials of the LR in Figure 6.2a above, the evaluation LC tends to oscillate between 0.3 and 0.4 across all the models. This could be attributed to the fact that downsampling introduces variance to the training.
If we observe the top performing models out of a total of 20 trials for the DNN in Figure 6.2b above, it marginally outperforms the LR models with the top model achieving an accuracy of 0.44. The shape of the LC do not exhibit any particular commonality with the partial exception of trial 20 and 1 with both decreasing in accuracy with the eventual increase.
Overall, an improvement from Scenario 1.

Scenario 3 : Cost Sensitive Training

Below are the results of the hyper-parameter training of 5000 training samples on 3 epochs.

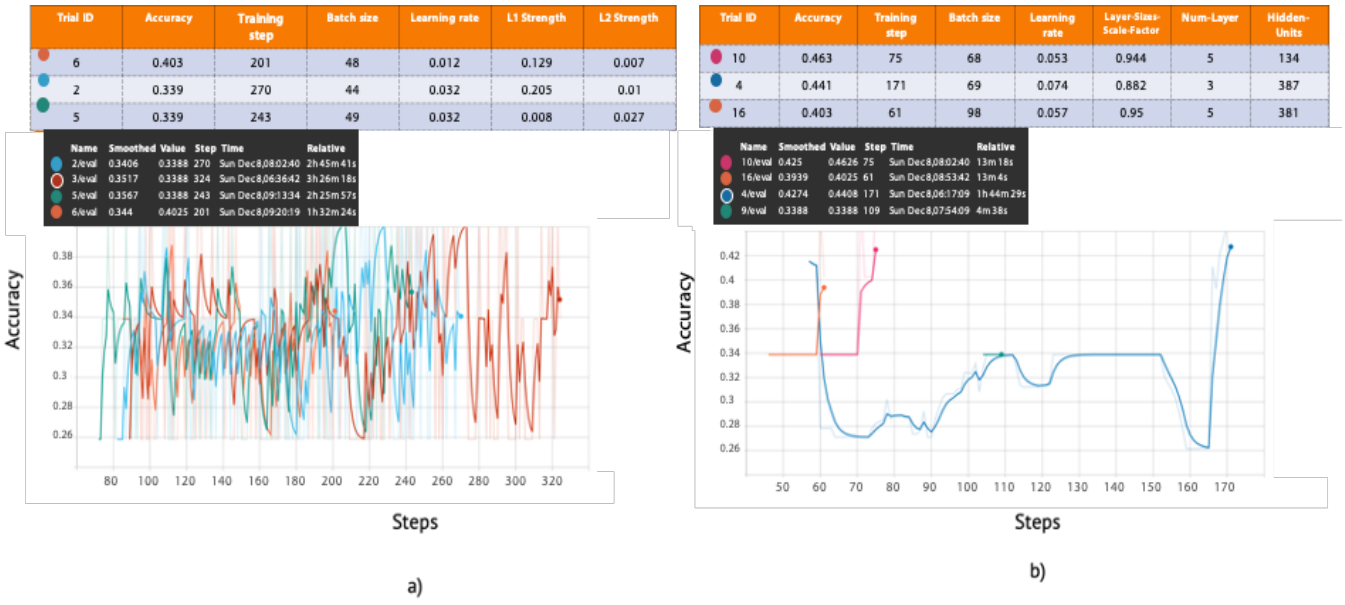

Figure 6.3: Scenario 3 : Top models and their parameters based on evaluation accuracy as a function of the training steps a) LR b) DNN

If we observe the top performing models out of a total of 20 trials of the LR in Figure 6.3a above, from a performance perspective , there is a general improvement in comparison to the other two scenarios (6.1, 6.1) with the best performing model obtaining an accuracy of 0,4. The LC tends to be far more erratic than a comparable shape in Scenario 1 .
If we observe the top performing models out of a total of 20 trials for the DNN in Figure 6.3b above, the models generally outperformed the LR counterpart and all the other DNN models in the other scenarios.
Overall, this scenario tends to provide superior results and will hence be used for further training on the full training data-set in Experiment 2.

6.2 Experiment 2 : Final Evaluation of Models on Test Set

With Scenario 3 (Cost-Sensitive Training) producing the best results, we will proceed with its training method across the full training data-set with the Scenarios' accompanying model parameters summarised in Table 6.2 below.

Parameter	Description	Logistic Regression	DNN
batch-size	Batch size for each training and evaluation step	48	68
learning-rate	Learning rate value for the optimizers	0.012	0.053
layer-sizes-scale-factor	Determine how the size of the layers in the DNN decays	-	0.944
num-layers	Number of layers in the DNN	-	5
hidden-units	Size of the first Hidden Layer	-	400
l1-strength	L1 Regularization Strength	0.129	-
l2-strength	L2 Regularization Strength	0.007	-

Table 6.2: Final parameters for DNN and LR models.

Once the training is complete, the final evaluation is applied to an out of sample test set. The evaluation results are presented in Tables 6.3 and 6.4 below.

6.2.1 Confusion Matrix Results

The results of the test set can be viewed in the form confusion matrix in Table 6.3 below

LR

		Predicted		
		Sig +	Insig	Sig -
Actual	Sig +	0	132	0
	Insig	0	3763	0
	Sig -	0	93	0

a)

DNN

		Predicted		
		Sig +	Insig	Sig -
Actual	Sig +	0	125	0
	Insig	377	3386	0
	Sig -	5	88	0

b)

Table 6.3: Confusion Matrix for LR and DNN models

What we can see from the confusion matrix in Table 6.3 in the context of the LR model is that even with the cost function training, the model is still biased towards the overweight Insig class as it only predicted this class.

When implemented in an operational setting, this would imply that no trades would be made. This is a result of overfitting.
What we can see from the confusion matrix in Table 6.3 in the context of the DNN model is that in contrast to the LR model, the DNN model did attempt to predict a label other than Insig with Sig + and unfortunately it did not attempt to classify the other minority class Sig - . This could imply that there are no distinguishing features in the Sig - class that the model could not identify. Overall, both models poorly classified.

6.2.2 Evaluation Metrics Results

The evaluation metrics results of the test set can be viewed in the Table 6.4 below

	LR				DNN			
	Precision	**Recall**	**F1-score**	**Support**	**Precision**	**Recall**	**F1-score**	**Support**
Sig -	0.00	0.00	0.00	93	0.00	0.00	0.00	93
Insig	0.94	1.00	0.97	3763	0.94	0.90	0.92	3763
Sig +	0.00	0.00	0.00	132	0.02	0.05	0.03	132
Micro avg	0.94	0.94	0.94	3988	0.85	0.85	0.85	3988
Weighted avg	0.89	0.94	0.92	3988	0.89	0.85	0.87	3988
Macro avg	0.32	0.33	0.32	3988	0.32	0.32	0.32	3988
	a)				b)			

Table 6.4: Results of test Set for the LR and DNN models

Precision and Recall per class

Both the DNN and LR models scored poorly on precision and recall for the minority classes (Sig +, Sig -) with the DNN slightly outperforming the LR with the attempt to classify the Sig + class. Both the DNN and LR models scored highly for the majority class (Insig) as anticipated.

F1 score per class

As a result of the poor precision and recall values for minority classes for both the DNN and LR models, as discussed in Section 6.2.2, the DNN and LR therefore scored poorly for the F1 score for the minority classes and scored well in the majority class.

Overall Performance

What we can observe, in the Micro avg and Weighted avg F1 in comparison to the Macro avg, as explained in Section 5.5.1, is that it shows an optically higher metric for the LR in comparison to the DNN due to its bias to the majority class. However if we observe the more suitable Macro avg F1 metric, that they generally fared the same. This illustrates the importance of observing your selected metrics in conjunction with the confusion matrix as we observed the slight outperformance of the DNN due to its attempt to classify one of the minority classes of which the LR failed to do so.

6.3 Experiment 3 : Further Evaluation of LR Model on Test Set

With the subpar results of Experiment 2 in mind, this Experiment looks to understand the potential contributors to these results by doing the following:

1. Focusing purely on LR models.
2. Determining whether the textual feature benefits the model performance in any way by having one model with only the textual feature (*sens_content*) and another model excluding the textual feature.

The model training will take the optimal parameters of Experiment 2 with the exception of the *L1 regularisation strength* as this will tend to eliminate irrelevant features as this could potentially be prevalent in the 512-dimensional textual feature. Hyperparameter tuning on the *L1 regularisation strength* was then conducted for the two models on a subset of the traning data for at least 4 trials (see Appendix A for results per trial) and the final parameters can be viewed in Table 6.5 below

Parameter	Description	LR- Text Feature Only	LR- Excluding Text Feature
batch-size	Batch size for each training and evaluation step	48	48
learning-rate	Learning rate value for the optimizers	0.012	0.012
l1-strength	L1 Regularization Strength	0.284	0.317
l2-strength	L2 Regularization Strength	0.007	0.007

Table 6.5: Final parameters for the two LR models.

Once the final parameters have been selected, the models are then run on 3000 training steps, just over 1 epoch, of the training data. Once the training is complete, the final evaluation is applied to an out of sample test set. The evaluation results are presented in Tables 6.6 and 6.7 below.

6.3.1 Confusion Matrix Results

The results of the test set can be viewed in the form confusion matrix in Table 6.6 below

LR – Text Feature Only

		Predicted		
		Sig +	Insig	Sig -
Actual	Sig +	132	0	0
	Insig	3763	0	0
	Sig -	93	0	0

a)

LR – Excluding Text Feature

		Predicted		
		Sig +	Insig	Sig -
Actual	Sig +	132	0	0
	Insig	3763	0	0
	Sig -	93	0	0

b)

Table 6.6: Confusion Matrix for the two LR models

What we can see from the confusion matrix in Table 6.6 is that in contrast to Experiment 2, both classifiers have their bias from the majority class Insig to one of the minority classes Sig +. When implemented in an operational setting, this would imply making solely buy trades on every event which can prove to be catastrophic in cases where the event should be a sell (Sig -). What can be deduced from this is that the highly non-linear relationships of the current feature representations, whether textual or not, and the class labels are inadequate for the simplistic LR model. This is prevalent in this Experiment as well as for the LR in Experiment 2 and as a result the classifier only targets one class and has no ability to distinguish the nuanced intricacies of the various events.

6.3.2 Evaluation Metrics Results

The evaluation metrics results of the test set can be viewed in the Table 6.7 below

	LR – Text Feature Only				LR – Excluding Text Feature			
	Precision	**Recall**	**F1-score**	**Support**	**Precision**	**Recall**	**F1-score**	**Support**
Sig -	0.00	0.00	0.00	93	0.00	0.00	0.00	93
Insig	0.00	0.00	0.00	3763	0.00	0.00	0.00	3763
Sig +	**0.03**	**1.00**	**0.06**	132	**0.03**	**1.00**	**0.06**	132
Micro avg	**0.03**	**0.03**	**0.03**	3988	**0.03**	**0.03**	**0.03**	3988
Weighted avg	0.00	**0.03**	0.00	3988	0.00	**0.03**	0.00	3988
Macro avg	**0.01**	0.33	**0.02**	3988	**0.01**	0.33	**0.02**	3988
	a)				b)			

Table 6.7: Results of test set for the two LR models

Precision and Recall per class

Consistent with the results observed in the confusion matrix in Table 6.7, the precision and recall values scored poorly across all the classes for both LR models with the exception of the perfect score for $Recall_{Sig+}$ as a result of the models only classiying Sig +.

F1 score per class

As a result of the poor precision and recall values for the Sig - and Insig classes for both LR models,the classifiers therefore scored poorly for the F1 score across the classes with a marginal difference in the Sig + class.

Overall Performance

What can be deduced from the subpar performnce from this Experiment as well as from Experiment 2 is the following:

- The poor relationships observed with the features and the labels, as alluded to in Section 3.4, are prevalent in the model performances and the utilisation of computational resources and model complexity cannot outweigh the importance of quality data with strong correlations with its labels.

Chapter 7

Conclusion and Future Works

In this final chapter, we conclude the theoretical study and analysis of this thesis. The main contributions of this project are discussed. Followed by the limitations and future work suggestions end off this chapter.

7.1 Conclusions

In this thesis, the importance of financial textual data used in NLFF was investigated in accordance with the goals outlined in the introduction, to reiterate, these goals were to:

1. *Better understand the relationship between information dissemination and stock price trends in the SA markets and*

2. *Utilise the relationship to forecast stock price trends with the eventual goal of assisting market participants in their trading decisions.*

A deep learning model DNN was adapted to the stock prediction which was compared with a classic machine learning model (LR). Upon reviewing the NLP literature in the context of NLFF and comparing the various information retrieval methods, the preferred pre-processing methods were implemented. The theoretical backgrounds of universal sentence encoder, event study and BO methods were reviewed. This is followed by discussions on

the implementation methods used for the prediction models. Overall, the following tasks were successfully designed and implemented:

- A web scraping algorithm to extract 7 years worth of SENS announcements as well as the associative features.
- A novel textual preprocessing and representation technique using a pretrained sentence encoder.
- A set of financial features for each announcement.
- Labelling each announcement by applying the event study methodology.
- The Logistic Regression (LR) based prediction model.
- The Deep Neural Network (DNN) based prediction model.
- An operationalised pipeline to use SENS announcements to predict the trend of stock prices.
- A series of relevant performance evaluation metrics.
- Running experiments to determine the values of optimised parameters for each model, taking into account the imbalanced nature of the data.

During the experiments, the performance metrics were used to evaluate the performance of the models in the following aspects:

- The prediction accuracy of each model given the 3 standard methods to handle class imbalance.
- The precision and recall of price trend prediction of each model on a per class basis.
- The **F1** score of price trend prediction of each model on a per class and overall basis.
- The comparison of the deep learning model and the classical model with the previous defined performance metrics.

Based on the implementation process and the experiment evaluation results, the following conclusion can be drawn:

- Cost sensitive training proved to be the superior method to handle class imbalance as compared to the other two methods investigated.
- Both the DNN and LR models poorly classified the minority classes as observed in the poor precision and recall values, with the DNN marginally better as it attempted to classify one of the minority classes while the LR completely biasing towards the majority class.
- With the applied features, both textual and market, it has been difficult to observe a strong relationship between information dissemination and stock price trends, which is evident with the sub-par model performances.

7.2 Final Remarks

Conducting descriptive analysis on the textual and market data, followed by a series of experiments observing a selection of performance metrics, we showed that *information dissemination has the potential to assist market participants with their trading decisions.* The comparison between the performance results of the two models showed that *deep learning models have a potential applicability in the context of stock trend prediction.* However, there are some limitations in this work, which are the following:

- Although we have demonstrated that incorporating information dissemination in the form of SENS announcements has the potential to assist market participants in their trading decisions, we do not claim that the methods and the implemented pipeline in this thesis can constitute as a viable trading strategy as further practical implications need to be considered such as the transaction and borrowing costs as well as slippage, to name a few.
- The ML framework used in this thesis, although robust, provided a limited option of multi-class models.

7.3 Future Work Directions

There are numerous avenues to extend this thesis further, here are a few recommendations based on learned lessons for future improvements:

- As with the availability of the US 8-K and 10-K/Q filings have sparked research, as evident in the literature reviewed in Section 2.1.1. So too we hope with this dataset being made readily available[1], that this will spark more research interest from various domains from Finance through to Data Science. In addition, the further accumulation of such data will be vital given the imbalanced nature of this data in order to obtain sufficient minority events (Sig +, Sig -) for effective model training.

- In this thesis, we only considered *market capitalisation* as the only market data feature. With the inclusion of additional market related features could potentially improve the models performance.

- There is an abundance of value to be derived from the textual features presented in this thesis that was not fully capitalised due to time constraints. By conducting unsupervised learning methods like clustering that can provide insight on how to improve feature engineering in supervised learning.

- The standardized 512-dimensional output for textual feature representation may have proved to be extensive and given time constraints other sentence embedding models with varying methods and output dimensions could not be investigated. Consider investigating other sentence embedding models such as the InferSent[2].

- As discussed in the limitations, there is an opportunity to outperform the models used in this thesis by implementing a more native class imbalance suitable models such as Decision Tree Based models like Random Forest and Gradient Boosting models.

[1] https://www.kaggle.com/katendencies/jse-sens-announcements
[2] https://github.com/facebookresearch/InferSent

Bibliography

Markowitz, Harry (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91.

Cowan, Arnold Richard (1992). "Nonparametric event study tests". In: *Review of Quantitative Finance and accounting* 2.4, pp. 343–358.

Page, Michael J and CV Way (1992). "Stock market over-reaction: The South African evidence". In: *Investment Analysts Journal* 21.36, pp. 35–49.

Schumann, Matthias and T Lohrbach (1993). "Comparing artificial neural networks with statistical methods within the field of stock market prediction". In: *[1993] Proceedings of the Twenty-sixth Hawaii International Conference on System Sciences*. Vol. 4. IEEE, pp. 597–606.

Henn, J and EvdM Smit (1997). "The influence of economic news events on share market activity in South Africa". In: *Investment Analysts Journal* 26.46, pp. 23–34.

Bhana, N (1998). "The share price reaction on the Johannesburg Stock Exchange for special (extra) dividend announcements". In: *Investment Analysts Journal* 27.47, pp. 5–15.

Wuthrich, Beat et al. (1998). "Daily stock market forecast from textual web data". In: *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*. Vol. 3. IEEE, pp. 2720–2725.

Bhana, N (1999). "The impact of public news regarding potential take-overs on the share price behaviour of target companies". In: *Investment Analysts Journal* 28.50, pp. 29–41.

Muller, C (1999). "Investor overreaction on the Johannesburg stock exchange". In: *Investment Analysts Journal* 28.49, pp. 5–17.

Tay, Francis EH and Lijuan Cao (2001). "Application of support vector machines in financial time series forecasting". In: *omega* 29.4, pp. 309–317.

Joachims, Thorsten (2002). *Learning to classify text using support vector machines*. Vol. 668. Springer Science & Business Media.

Peramunetilleke, Desh and Raymond K Wong (2002). "Currency exchange rate forecasting from news headlines". In: *Australian Computer Science Communications* 24.2, pp. 131–139.

Fung, G Pui Cheong, J Xu Yu, and Wai Lam (2003). "Stock prediction: Integrating text mining approach using real-time news". In: *Computational Intelligence for Financial Engineering, 2003. Proceedings. 2003 IEEE International Conference on*. IEEE, pp. 395–402.

Antweiler, Werner and Murray Z Frank (2004). "Is all that talk just noise? The information content of internet stock message boards". In: *The Journal of finance* 59.3, pp. 1259–1294.

Mittermayer, M-A (2004). "Forecasting intraday stock price trends with text mining techniques". In: *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*. IEEE, 10–pp.

Li, Feng et al. (2006). *Do stock market investors understand the risk sentiment of corporate annual reports*.

Rachlin, Gil et al. (2007). "ADMIRAL: A data mining based financial trading system". In: *2007 ieee symposium on computational intelligence and data mining*. IEEE, pp. 720–725.

Soni, Ankit, Nees Jan van Eck, and Uzay Kaymak (2007). "Prediction of stock price movements based on concept map information". In: *2007 IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making*. IEEE, pp. 205–211.

Tetlock, Paul C (2007). "Giving content to investor sentiment: The role of media in the stock market". In: *The Journal of finance* 62.3, pp. 1139–1168.

Zhai, Yuzheng, Arthur Hsu, and Saman K Halgamuge (2007). "Combining news and technical indicators in daily stock price trends prediction". In: *International symposium on neural networks*. Springer, pp. 1087–1096.

Feldman, Ronen et al. (2008). "The incremental information content of tone change in management discussion and analysis". In:

Henry, Elaine (2008). "Are investors influenced by how earnings press releases are written?" In: *The Journal of Business Communication (1973)* 45.4, pp. 363–407.

Tetlock, Paul C, Maytal Saar-Tsechansky, and Sofus Macskassy (2008). "More than words: Quantifying language to measure firms' fundamentals". In: *The Journal of Finance* 63.3, pp. 1437–1467.

Butler, Matthew and Vlado Kešelj (2009). "Financial forecasting using character n-gram analysis and readability scores of annual reports". In: *Canadian Conference on Artificial Intelligence*. Springer, pp. 39–51.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Li, Feng (2009). "The determinants and information content of the forward-looking statements in corporate filings-a Naıve Bayesian machine learning approach". In: AAA.

Schumaker, Robert P and Hsinchun Chen (2009). "Textual analysis of stock market prediction using breaking financial news: The AZFin text system". In: *ACM Transactions on Information Systems (TOIS)* 27.2, p. 12.

Sun, Yanmin, Andrew KC Wong, and Mohamed S Kamel (2009). "Classification of imbalanced data: A review". In: *International Journal of Pattern Recognition and Artificial Intelligence* 23.04, pp. 687–719.

Brochu, Eric, Vlad M Cora, and Nando De Freitas (2010). "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning". In: *arXiv preprint arXiv:1012.2599*.

Huang, Chenn-Jung et al. (2010). "Realization of a news dissemination agent based on weighted association rules and text mining techniques". In: *Expert Systems with Applications* 37.9, pp. 6409–6413.

Li, Feng (2010). "The information content of forward-looking statements in corporate filings—A naıve Bayesian machine learning approach". In: *Journal of Accounting Research* 48.5, pp. 1049–1102.

Ward, Mike and Chris Muller (2010). "The long-term share price reaction to black economic empowerment announcements on the JSE". In: *Investment Analysts Journal* 39.71, pp. 27–36.

Groth, Sven S and Jan Muntermann (2011). "An intraday market risk management approach based on textual analysis". In: *Decision Support Systems* 50.4, pp. 680–691.

Loughran, Tim and Bill McDonald (2011). "When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks". In: *The Journal of Finance* 66.1, pp. 35–65.

Watermeyer, Renen (2011). "The JSE Stock Exchange News Service: the impact of SENS announcements on trading activity on the JSE securities exchange". PhD thesis. University of Cape Town.

Bergstra, James and Yoshua Bengio (2012). "Random search for hyper-parameter optimization". In: *Journal of machine learning research* 13.Feb, pp. 281–305.

Engelberg, Joseph E, Adam V Reed, and Matthew C Ringgenberg (2012). "How are shorts informed?: Short sellers, news, and information processing". In: *Journal of Financial Economics* 105.2, pp. 260–278.

Schumaker, Robert P, Yulei Zhang, et al. (2012). "Evaluating sentiment in financial news articles". In: *Decision Support Systems* 53.3, pp. 458–464.

Vu, Tien Thanh et al. (2012). "An experiment in integrating sentiment features for tech stock prediction in twitter". In: *Proceedings of the workshop on information extraction and entity analytics on social media data*, pp. 23–38.

Hagenau, Michael, Michael Liebmann, and Dirk Neumann (2013). "Automated news reading: Stock price prediction based on financial news using context-capturing features". In: *Decision Support Systems* 55.3, pp. 685–697.

Jin, Fang et al. (2013). "Forex-foreteller: Currency trend modeling using news articles". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 1470–1473.

Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Vapnik, Vladimir (2013). *The nature of statistical learning theory*. Springer science & business media.

Yang, Lili et al. (2013). "Combining lexical and semantic features for short text classification". In: *Procedia Computer Science* 22, pp. 78–86.

Yu, Yang, Wenjing Duan, and Qing Cao (2013). "The impact of social and conventional media on firm equity value: A sentiment analysis approach". In: *Decision Support Systems* 55.4, pp. 919–926.

Kearney, Colm and Sha Liu (2014). "Textual sentiment in finance: A survey of methods and models". In: *International Review of Financial Analysis* 33, pp. 171–185. ISSN: 1057-5219. DOI: https://doi.org/10.1016/j.irfa.2014.02.006. URL: http://www.sciencedirect.com/science/article/pii/S1057521914000295.

Lee, Heeyoung et al. (2014). "On the Importance of Text Analysis for Stock Price Prediction." In: *LREC*, pp. 1170–1175.

Li, Qing, TieJun Wang, et al. (2014). "The effect of news and public mood on stock movements". In: *Information Sciences* 278, pp. 826–840.

Li, Qing, Tiejun Wang, et al. (2014). "Media-aware quantitative trading based on public web information". In: *Decision support systems* 61, pp. 93–105.

Li, Xiaodong et al. (2014). "Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information". In: *Neurocomputing* 142, pp. 228–238.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Quinlan, J Ross (2014). *C4. 5: programs for machine learning*. Elsevier.

Sidorov, Grigori et al. (2014). "Syntactic n-grams as machine learning features for natural language processing". In: *Expert Systems with Applications* 41.3, pp. 853–860.

Henry, Elaine and Andrew J Leone (2015). "Measuring qualitative information in capital markets research: Comparison of alternative methodologies to measure disclosure tone". In: *The Accounting Review* 91.1, pp. 153–178.

Iyyer, Mohit et al. (2015). "Deep unordered composition rivals syntactic methods for text classification". In: *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pp. 1681–1691.

Peng, Yangtuo and Hui Jiang (2015). "Leverage financial news to predict stock price movements using word embeddings and deep neural networks". In: *arXiv preprint arXiv:1506.07220*.

Weiss, Sholom M, Nitin Indurkhya, and Tong Zhang (2015). *Fundamentals of predictive text mining*. Springer.

Huynh, Huy D, L Minh Dang, and Duc Duong (2017). "A new model for stock price movements prediction using deep neural network". In: *Proceedings of the Eighth International Symposium on Information and Communication Technology*. ACM, pp. 57–62.

Jothimani, Dhanya et al. (2017). "Ensemble of non-classical decomposition models and machine learning models for stock index prediction". In: *MWAIS 2017 Proceedings* 17, pp. 1–5.

Kraus, Mathias and Stefan Feuerriegel (2017). "Decision support from financial disclosures with deep neural networks and transfer learning". In: *Decision Support Systems* 104, pp. 38–48.

Cer, Daniel et al. (2018). "Universal Sentence Encoder". In: *CoRR* abs/1803.11175. arXiv: 1803.11175. URL: http://arxiv.org/abs/1803.11175.

Feuerriegel, Stefan and Julius Gordon (2018). "Long-term stock index forecasting based on text mining of regulatory disclosures". In: *Decision Support Systems* 112, pp. 88–97.

Xing, Frank Z, Erik Cambria, and Roy E Welsch (2018). "Natural language based financial forecasting: a survey". In: *Artificial Intelligence Review* 50.1, pp. 49–73.

Announcement Types and Announcement Sub Types (n.d.). URL: https://sens.jse.co.za/webhelp/webhelpeu/announcement_types.htm.

Appendix A

Experiment 3: Hyperparameter Tuning Results

HyperTune trials

Trial ID	f1 ↓	Training step	Elapsed time	l1-strength
5	0.02149	60	3 hr 30 min	0.28423
2	0.00514	60	3 hr 54 min	0.31716
4	0.00228	60	3 hr 55 min	0.13785
3	0.00228	60	3 hr 25 min	0.11366
1	0.00222	60	3 hr 56 min	0.5

Table A.1: Hyperparameter Tuning Results for LR with only textual feature

HyperTune trials

Trial ID	f1 ↓	Training step	Elapsed time	l1-strength
4	0.90402	60	7 min 17 sec	0.1
3	0.90402	60	6 min 46 sec	0.7035
2	0.90402	60	7 min 16 sec	0.31716
1	0.90402	60	6 min 38 sec	0.5

Table A.2: Hyperparameter Tuning Results for LR with excluding textual feature