Imperial College London
Department of Computing

Multi-View Learning Approaches for Stock Return Prediction with Tweets

Author:
Karolina Marta Sowinska

Supervisor:
Dr Pranava Madhyastha

Submitted in partial fulfilment of the requirements for the
MSc Degree in Computing Science of Imperial College London
September 2019

Abstract

The purpose of this project is to investigate the effect of semantic information contained in tweets on one-, two-, three- and seven-day stock return prediction accuracy. We also explore if this accuracy is improved by fusing together multiple kinds of features in the multi-view learning process. In order to answer the posed questions, we create an original company-level labelled dataset with 862,231 tweets which we release for the NLP community to use. We then proceed to extracting features from the tweets using multiple methods, including CountVectorizer, fastText, BERT. To our best knowledge, we are the first study that treats these various kinds of features as multiple "views" of the same text for the purposes of the stock return prediction. We use Canonical Correlation Analysis (CCA) decomposition to project the feature vectors into a new feature space where they are highly correlated. The resulting vectors are concatenated, becoming the input for a binary classification algorithm predicting upward or downward movement of the price. We employ multiple machine learning techniques for this task, namely Feedforward Neural Networks, SVM, Gradient Boosting, auto-sklearn. We obtain 67% prediction accuracy for a seven-day return with a combination of BERT and fastText features using auto-sklearn algorithm. We find evidence that multi-view model outperforms single-view benchmarks, but overall textual information does not increase the stock return prediction accuracy.

Acknowledgements

I owe a debt of gratitude to my supervisor, Dr Pranava Madhyastha at Imperial College London. From helping me to formulate the hypothesis, through directing me towards the state-of-the-art Natural Language Processing techniques, to continually reviewing my progress, he has had a pivotal contribution to this project. I thank him not only for generously sharing his expertise, but also for having patience for the person who only recently began her journey in the field of Computer Science.

I would also like to acknowledge Dr Julio Amador Diaz Lopez at Imperial College London as the second reader of this thesis, and I am thankful for his valuable comments on this project.

Moreover, I am grateful to the team of experts at Schroders who agreed to meet with me to discuss the commercial applicability of this project.

Lastly, I would like to thank my family. Dziekuje moim rodzicom i Adrianowi za wsparcie przez wszystkie lata wyzszej edukacji. Mialam ogromne szczescie otaczac sie ludzmi, ktorzy wierzyli we mnie nawet bardziej, niz ja sama wierzylam.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivations

Traditionally investors used to build their stock return prediction models based exclusively on numerical data. They would then refer to qualitative information like news articles to finalise their conviction about a given investment idea. However, this process can be hugely intuitive and error-prone. As Andrew Lo mentioned in Adaptive Markets, *we are not homo economicus, but homo sapiens*. The author suggests that we cannot process the vast amounts of data or even make rational decisions based on our analysis. These inherent human flaws could be eliminated from the investment process by means of painstakingly objective machine learning algorithms. Nevertheless, building a model that mimics the human approach to investment analysis is an overly ambitious task. Although psychological biases can be thought of as weaknesses of human thought, they can also be viewed as revealing something profound about the complexity of human cognition. Investors' decisions are based on a rich understanding of the companies they are investing in, the broader environment those companies operate in (including competitors, governments, regulators, etc.), and an understanding of the factors that influence the market behaviour. Replicating the decisions made by investors would therefore require a researcher to create Artificial General Intelligence, which, arguably, is out of scope of a summer project.

Having discarded the algorithmic absolutism, we would like to take a pragmatic stance on the

topic. Machine learning techniques should be used as a leverage; assisting investors in making the right decisions, rather than replacing human thought. Thus, we would like to build a model which processes the abundance of qualitative data from social media posts - a task which might be too time-consuming to be viable manually - and suggests whether the textual information predicts the upward or downward movement in the stock price over the following week. This model has a practical application for a buy-and-hold investor. Before the investor executes a trade, he wants to make sure that he will buy the stock at the lowest possible price. Thus, understanding if the stock price of a particular company will increase or decrease over the next couple of days is cost-effective, especially when dealing with large transaction sizes.

Despite the numerous research efforts to predict the stock market returns with Twitter posts, returns of individual companies are typically not taken into account. The most popular studies in the field investigate the stock market movement as a whole (Bollen et al. n.d.), (Li et al. 2014), (Oliveira et al. 2017). The problem with such an approach is that aggregate models are not useful for portfolio managers who are interested in predicting the stock price of a particular company. Thus, we construct a dataset that allows for company-level analysis of Twitter sentiment impact on one-, two-, three-, and seven-day stock returns. We aim to release the dataset for the NLP community in support of investigations in language sciences.

Analysing natural text is a complex task. There exist multiple techniques to turn the text into numbers (so called features) that can be fed into machine learning models, e.g. Count Vectorizer, fastText, BERT. Each feature extraction method allows us to capture different kind of semantics from the text. While Count Vectorizer is excellent at recording the frequencies of words in a tweet, it is unable to derive contextual information. For example, the word "shelf" will be represented in the same way in both of these sentences: "Ice shelf is melting", "I put a book on a shelf". However, BERT algorithm would allow us to capture the context of each word, given the words appearing both to the left and to the right of "shelf". We believe that superior results can be achieved by investigating multiple types of features simultaneously. Thus, we propose multi-view learning approach, in which we look at the same text from multiple "views" by means of multiple feature types. As far as we know, no previous research has investigated this approach for stock return prediction.

1.2 Objectives

The project has two main objectives. Firstly, it aims to answer two research questions:

1. Does the semantic information in tweets improve the benchmark accuracy of one-, two-, three- and seven-day stock return predictions?
2. Do multiple kinds of features fused together in the multi-view learning process improve the accuracy of one-, two-, three- and seven-day stock return predictions?

Secondly, we aim to generate an original dataset from raw Twitter data which allows for company-level analysis of tweets for one-, two-, three-, and seven-day stock return prediction. The ultimate goal is to produce a labelled dataset that can allow both us, and the entire Natural Language Processing community to investigate the posed research questions.

1.3 Contributions

We generate a labelled dataset in two versions:

1. **Full dataset** (filtered by company names and company usernames on Twitter) - **862,231** samples
2. **Reduced dataset** (filtered by company usernames on Twitter) - **85,176** samples

The dataset includes tweets that mention at least one of the 100 selected companies. The companies were chosen according to the ranking Financial Times' Top 100 Global Brands 2019 ranking. The tweets are collected between July and October 2018. However, this is not a time-series data, as there is no guarantee that each company is mentioned in tweets at least once a day. Instead, we make an assumption that every day for any company is an independent event. In the context of the sentiment analysis, we believe that this assumption is likely to hold true. We elaborate on this idea in 3.3.3. The dataset has four labels: one-, two-, three- and seven-day

stock return. This is calculated as the percentage difference between the price on the tweet's date of creation and the future price with a specified lag. The dataset also includes the following five metrics: Trading Volume, 10 Day Volatility, 30 Day Volatility, LSTM polarity, TextBlob polarity. The dataset used in this study was expanded with Bloomberg's proprietary metrics, which we include in our benchmarks: News Sentiment Daily Average, News Heat Publication Daily Average, Twitter Publication Count, Twitter Positive Sentiment Count, Twitter Negative Sentiment Count. However, they cannot be released under the Bloomberg's Privacy Policy.

We build a multi-view learning model using Canonical Correlation Analysis decomposition. Using auto-sklearn, which is an award-winning automated machine learning library, we achieve 67% prediction accuracy on the reduced dataset for the seven-day return with fastText and BERT features. This result is largely in line with previous findings, marginally outperforming coupled matrix and tensor factorization scheme (Zhang et al. 2018) and SMeDA-SA (system generating a concept map for the target concept based on other concepts that are found to be associated with it) (Li et al. 2014) for companies that are not grouped by industry. Our model outperforms the algorithm from the first study by 0.52 percentage points, and the second study by 4.5 percentage points. However, a direct comparison cannot be made as those studies tested their models on different datasets.

The rest of this report is organised as follows. Chapter 2 delineates the previous statistical and machine learning approaches to stock return prediction, notably Natural Language Processing techniques in 2.5 and Multi-view learning in 2.6. Chapter 3 describes the process of dataset generation - collection of raw data as well as merging the Twitter and Bloomberg datasets. In Chapter 4 we outline the process of feature and polarity extraction. In Chapter 5 we scrutinise the resulting dataset. In addition to vocabulary analysis, we try to derive and understand the main themes in the data by means of topic modelling and text clustering. Chapter 6 reports the results from our multi-view model. In this chapter we also evaluate our results against the literature findings in 6.5. Finally, we conclude our efforts in Chapter 7.

Chapter 2

Background Theory

2.1 Introduction

In this Chapter we aim to position our project within the frame of existing body of literature on the stock return prediction. Firstly, we discuss the neoclassical theory of stock prediction, notably Efficient Market Hypothesis. It states that stock prices cannot be predicted, because markets are informationally efficient. However, comforted with a series of empirical studies that discredit the theory, we proceed to analysing machine learning approaches to stock return prediction. Next, we focus specifically on the previous studies within the realm of Natural Language Processing. Lastly, we discuss the previous multi-view learning approaches for stock return prediction.

2.2 The neoclassical theory of stock prediction

Discussions regarding the stock returns predictability have dominated the financial research in recent years. The Efficient Market Hypothesis, whose development was honoured with a Nobel Prize in Economics in 2013, advocates that stock returns cannot be predicted. The theory contends that stock prices fully reflect all available information about the market at any given

time (Fama 1970). This implies that investors cannot predict future stock prices on the basis of the current information, because all the information is already accounted for in the current price. The ground to this theory was given by the prevailing belief in economics that investors are rational agents. Thus, they instantly react to any new information signal to take advantage of arbitrage opportunities (i.e. differences in prices of the same good) and maximise profits as a result. This constant trading causes the stock prices to move to incorporate all the public information in a process known as the price discovery (Campbell et al. 1997). If this theory held, all efforts to predict stock returns would be futile.

However, any observable trends in stock returns imply a certain degree of predictability, which suggests that markets are not always efficient. The reasons for that are coming from a human nature: we are not always the rational agents who make perfectly optimal decisions. This psychological aspect of investing has propelled to the forefront in investigations of stock return predictability.

2.3 Statistical approaches to stock prediction

Several economic studies agree that stock prices can be predicted, even by means of simple statistical methods like linear regression. This is best illustrated by the study on accuracy of predictions for small stocks (Antoniou et al. 2012). The authors investigated the role of the psychological bias known as ambiguity aversion on predicting stock returns of small companies. The study shows that investors are systematically more pessimistic about small companies' returns or in some cases, the available information may be of such low quality that investors are unable to confidently estimate the distribution of expected earnings. Thus, after the quarterly earnings announcement, investors are often surprised by how well the company performed, and they rush to buy the stock. This creates abnormal positive returns around earning announcements. Thus, stock returns of small companies exhibit a pattern, which constitutes evidence against market efficiency.

A number of other authors have posited that Efficient Market Hypothesis should be refuted based on the results from their linear models. M.Cooper, W.Jackson and G.Patterson corrob-

orate this view in their study of the predictability of bank stock returns in the cross section. Their results suggest that investors under-react to changes in banks fundamental variables. Moreover, their out-of-sample testing demonstrates that this under-reaction is exploitable using simple cross-sectional trading strategies (Cooper et al. 2003). Another study investigated the predictive power of 36 different financial ratios on the Mexican stock market, predicting the return 4 years into the future. The regression model that the authors estimated takes the form:

$$r_{t+n} = \alpha + \beta_1 FR_t + \beta_2 log(MVE) + \beta_3 BETA_{t+n} + \epsilon \tag{2.1}$$

where r_{t+n} is stock return at quarter t + n (n is 4 for 1 year future returns), FR is a set of 36 most preferred financial ratios at quarter t, log (MVE) is the logarithm of the market capitalisation of the firm's equity, and BETA, the measure of systematic risk, which acts as a control variable. The study found all of the coefficients from the regression to be statistically significant at 1% significance level (Trejo Pech et al. 2015). This finding evidences the predictability of stock returns.

The academic community has explored the statistical approaches to stock return prediction with success. Thus, despite the accepted economic theory, which rejects such research as futile, the efforts to further improve the prediction by means of machine learning methods are fully justified.

2.4 Machine Learning approaches to stock prediction

The rising volume and availability of data has critically influenced academic dialogue on predicting the stock returns. Numerous machine learning methods have been investigated, notably multilayer perceptron, Support Vector Machines, case-based reasoning, decision trees, generalised rule induction and logistic regression (Patankar & Swati 2014). Chen and Zhang investigated the use of Support Vector Machines for stock prediction. As a data input they used financial ratios of companies listed on Shangai Stock Exchange. The authors created two models, one for lineary seperable training set, and the other for complicated, high-dimensional

financial ratios which show nonlinear relationships. The evaluation reveals that portfolios constructed with stocks selected by the model outperformed the benchmark, i.e. the A-share index of Shanghai Stock Exchange.(Yu et al. 2014). Prior research has also thoroughly investigated the applicability of neural networks in their various forms: Multi-Layered Feed Forward Network with back propagation algorithm (Mehrara et al. 2010), Adaptive Neuro-Fuzzy Inference System (Agrawal et al. 2010), Radial Basis Function Network and Recurrent Neural Network (Phua et al. 2003), Long Short Term Memory (as a specific type of RNN) (Roondiwala et al. 2015) to mention a few. However, most of these studies focus on forecasting the returns of the stock market as a whole, not individual stocks.

A study conducted by M.Abe and H.Nakayama from Nomura Asset Management in 2018 that trains a deep neural network with 25 financial ratios was found to be particularly relevant for this research, as it predicts returns for individual stocks. The paper creates a model to predict one-month-ahead stock returns in the cross-section in the Japanese stock market (319 companies that constitute MSCI Japan Index). The input comprises of the financial ratios, and the output is defined as the next month's stock return. The study has shown that better prediction accuracy is generally achieved with a greater number of layers (as tested on 3,5 and 8-layer architectures). A comparison was made between the deep neural networks and Support Vector Machines and Random Forest. Deep neural networks outperformed both of these methods. (Abe & Nakayama n.d.).

Prior research substantiates the belief that deep neural networks are a highly promising method for predicting future stock returns of individual stocks based on financial information. They were found to outperform other machine learning techniques such as SVM and Random Forest. Thus, we also decide to train neural networks for our prediction task.

2.5 Sentiment analysis for stock prediction

The advancements in Natural Language Processing have dramatically shaped queries on stock return prediction in recent years. Social media content and news articles reflect the moods and opinions that people have about certain companies, which in turn can be a predictor of the

company's stock returns. Such emotions can be described as a sentiment - a positive sentiment value captures messages with an overall positive tone, and negative sentiment value captures messages with an overall negative tone. The sentiment of textual data such as news articles or tweets can be detected and administered to investigations of the stock returns, often as a sole input to a model. One prominent study that made an attempt at this was conducted by (Li et al. 2017). The researcher developed a novel way called Social Media Data Analyzer – Sentiment Analysis (SMeDA-SA) for mining ambiguous temporal data, such as that contained in the Twitter posts. SMeDA-SA is conducted in several steps:

1. NLP techniques are applied to classify a tweet's sentiment into five categories (Positive +, Positive, Neutral, Negative, Negative -)

2. Target concepts that the user is interested in (e.g., a particular company, such as "Apple Inc.") are identified.

3. The method makes use of an association discover algorithm to discover all concepts that are associated with the target concept (e.g., the "MacBook Pro" can be discovered to be associated with "Apple Inc.")

4. The method uses an algorithm developed for semantic analysis to generate a concept map for the target concept (in our example, "Apple Inc") based on other concepts that are found to be associated with it

5. The degree of association is determined between the concept maps and the five sentiment categories

6. Finally, another algorithm is applied to see if there is correlation between stock prices and the sentiment detected in the tweets.

The researchers looked for the association between the sentiment measure and the daily movement of the stock price. The strength of this study is that it recognises the possible delay that the social media content might have on the stock price. Since the sentiment value may not be immediately reflected in the company's stock price, the study investigates multiple lags.

The study results show that, in general, for the 30 selected companies, the association between sentiment and the stock price seems to be strongest when the lag is set to T+3. This is an interesting finding which stands in opposition to the Efficient Market Hypothesis which assumes that all publicly available information is immediately reflected in the stock price. To evaluate the effectiveness of the proposed approach and accuracy of stock movement prediction based solely on social media data, the researchers performed experiments on 30 NASDAQ and NYSE listed companies. For mining social media data for public sentiment, they collected approximately 15 million records of Twitter data that mentioned these 30 companies either directly or indirectly by mentioning their products or services. The algorithm achieved the average prediction accuracy of 66.48%, which is the highest compared to other data mining algorithms (SVM, C4.5, Naive Bayesian).

Another study conducted by Johan Bollen and Huina Mao in 2010 earns our special attention as one of the most influential studies on Twitter sentiment analysis for stock return prediction, with almost 4000 citation according to Google Scholar. However, it is arguably less relevant to this project, because it investigates the stock market returns as a whole (rather than returns of individual stocks). The researchers study the relationship between daily sentiment of a collection of Tweet posts and Dow Jones Industrial Average (DJIA), which is a index indicating the overall movement of the stock market prices. Abstracting away from failure to distinguish between individual stocks, another weakness of this study is that the researchers do not detect the sentiment on their own. Instead, they rely on third-party tools, OpinionFinder and GPOMS. Thus, it is difficult to assess the adequacy and appropriateness of the applied sentiment analysis techniques. The authors describe OpinionFinder as "measuring positive vs. negative mood from text content", and GPOMS "measuring 6 different mood dimensions from text content". The researchers perform a Granger causality analysis in which they correlate DJIA values to two sentiment time series obtained from the sentiment tools. They also deploy a Self- Organising Fuzzy Neural Network model to test the hypothesis that the prediction accuracy of DJIA prediction models can be improved by including measurements of public mood. Forecasting accuracy is measured in terms of the average Mean Absolute Percentage Error (MAPE) and the direction accuracy (up or down). The study finds that the "calm" mood is

the most predictive of the direction of the stock market, with the 86.7% direction accuracy. Other investigated moods, namely "happy", "sure", "vital", "kind", "alert", do not have a similar predictive power. However, the study substantiates the idea that public mood (in this case the calmness of the public) can be predictive of the stock market movement, even as tested with simple, third-party, general sentiment analysis tools. Therein lies the phenomenon of this study's popularity - it reveals the clear relationship between the social media sentiment and stock returns.

2.6 Multi-view learning for stock prediction

A few studies employed multi-view learning to the sentiment analysis problem. The academic community distinguishes a number of representative multi-view learning algorithms in different areas and classifies them into three groups:

1. co-training
2. multiple kernel learning
3. subspace learning

Co-training style algorithms train alternately to maximise the consensus on two distinct views of the data, multiple kernel learning algorithms combine a few kernels either linearly or non-linearly, whereas subspace learning algorithms "aim to obtain a latent subspace shared by multiple views by assuming that the input views are generated from this latent subspace". (Xu & Xu 2013). The first and the second approach seem to be more frequently incorporated for the stock return prediction task. For example, (Li et al. 2014) combines the sentiment from market news and historical stock prices in order to improve prediction accuracy of stock returns. The authors employ multiple kernel learning in order to combine the two views, and their results outperform three baseline methods (which use only one information source or use a naive combination of the two sources) on test data. Similarly, (Shynkevich et al. 2016) use the same type of multi-view learning algorithm to combine information extracted from

multiple news categories for the prediction of healthcare stocks. Integrating the information from five different kernels gives superior results to single-view models in this study. The usage of co-training algorithms for the stock return prediction was noted in the study conducted by (Ben-Ami et al. n.d.). The authors first label the tweets into positive, neutral, and negative categories with a text-based Sentiment Analysis system. This system is assumed to classify tweets into positive and negative categories with a satisfactory precision, but insufficient recall. Thus, the tweets that fall into the neutral category cannot be assumed to contain only neutral sentiment. Multi-view learning is employed to solve this issue. The researchers recognised that they can utilise the meta-data of a tweet, as long as it is conditionally independent of the tweet's content given its polarity. Using this additional representation, the study trains a binary SVM with positive and negative tweets as the training data. The classifier then produces the sentiment for the neutral set of tweets. The results confirm again that, when multiple sources are combined, they produce much better accuracy than single-view data.
However, prior studies have failed to utilise the multi-view learning to combine the vectors from multiple feature extractors as separate views in order to improve the stock return prediction accuracy. Our project probes into this unexplored territory.

2.7 Conclusion

There exists a considerable body of literature on stock return prediction problem. Multiple studies have used the recent advanced in machine learning in order to improve the predictive accuracy of the models. We have seen that some of these studies turned to the Natural Language Processing field to investigate the impact of social media posts and news articles sentiment on the stock return. The previous academic research shows that sentiment analysis can be useful for the prediction task. This is also true for employing multi-view learning approaches. However, most studies focused on fusing heterogeneous information sources. To our best knowledge, we are the first study that considers various kinds of features as multiple "views" for the stock return prediction task.

Chapter 3

Dataset

3.1 Introduction

A common strategy used to investigate the impact of Twitter posts on stock returns is to consider the stock market movement as a whole. Thus, the returns of individual companies are typically not taken into account (Bollen et al. n.d.), (Li et al. 2014), (Oliveira et al. 2017). The problem with such an approach is that aggregate models are not useful for portfolio managers who are interested in predicting the stock price of a particular company. *We construct a dataset that allows for company-level analysis of Twitter sentiment's impact on one-, two-, three-, and seven-day stock returns.* Structuring a few terabytes of raw Twitter data is a risky endeavour, and we encounter multiple technical problems, notably arising from memory limitations. We combat these obstacles with efficient algorithms and partitioning the data into manageable parts. The ultimate goal is to produce a labelled dataset that can allow the Natural Language Processing community to investigate the aforementioned relationship. The resulting models with sufficient predictive accuracy will have the potential to assist portfolio managers in the decision making process of investing in stocks. Our research also aims at designing such predictive models for this challenging problem in Chapter 6.

3.2 Data Collection

3.2.1 Twitter Data Collection

Ready datasets

Despite the maturing of the Twitter sentiment analysis field, with a wealth of well-understood methods and algorithms, publicly available datasets for stock return prediction are scarce. The datasets that we did find were not suitable for answering the research question of this study, so we decided to create our own dataset.

Twitter Developer API

Twitter offers a Twitter API service which allows for searching through an archive of tweets dating back to March 2006. The API supports a single query of up to 128 chars per request in the free sandbox version. The query is also limited in returning a maximum 100 Tweets per request. In order to use the Twitter API, we set up a Twitter Developer account and registered an app called DeepLearningForInvesting (app ID: 16375308). However, due to the volume limitation, we found this approach insufficient for developing a robust dataset for stock return prediction.

Off-the-shelf Scrapers

In order to obtain a larger volume of Tweets, we also investigate multiple off-the-shelf tweet scraper providers. Notably, we investigate Tweepy and The Digital Methods Initiative Twitter Capture and Analysis Toolset (DMI-TCAT). Both tools allow for large volume download, however, they do not support queries which date further than 7 days back. Thus, this proved to be impractical for the purposes of this study.

Unstructured data download

Due to the aforementioned limitations of the Twitter API and the off-the-shelf scraper methods, we decide to download almost 2TB of unprocessed Tweets, and filter them using our custom scripts to create the desired dataset. The link to the unprocessed data download page was provided by a member of ResearchGate website. The data was provided in the "Spritzer" version, the most light and shallow of Twitter grabs, which captures a 1% random sample of public tweets. We collected tweets from between July and October 2018.

3.2.2 Investigated Stocks

We investigate tweets that mention one of the 100 stocks of public companies that own the most valuable brands, according to Financial Times' Top 100 Global Brands 2019 rank. We choose to investigate such companies, because we believe they elicit emotional responses in people, which is reflected in the sentiment of social media posts. Choosing to investigate stocks which constitute a specific share index (e.g. S&P 500) could result in less meaningful results, because despite their large market capitalisations, they can be less well known to the public. This is often the case with the parent companies with multiple divisions. For example, Whitbread PLC is not a household name, and we assume that it is unlikely to be tweeted about. Its subsidiaries - Costa Coffee and Premier Inn hotels are much more popular, but tweets relating to those names would not be captured when using the index approach. Thus, we believe that investigating valuable brands instead can lead to more meaningful results.

Some brands featuring in the ranking are owned by non-public companies, i.e. their shares are not traded on any stock exchange. An example of such a brand is Louis Vuitton, which is owned by Louis Vuitton Malletier, a private company. Such companies were not taken into account, because naturally they do not have a public share price.

3.2.3 Bloomberg Data Collection

The financial data used in this study was accessed from a Bloomberg terminal on Imperial College London's licence. For each of the 100 stocks we have collected the following financial data:

Table 3.1: Financial data collected from Bloomberg

Financial metric	**Description**
Last Price	Closing stock price
Volume	The amount of stock that was traded on a given day
10 Day Volatility	The range of price change a stock experienced over the last 10 days
30 Day Volatility	The range of price change a stock experienced over the last 30 days

The financial data shown in Table 3.1 was collected over the period 01.01.2017-01.07.2019 with a daily frequency. It is typical of financial data to contain missing values for Saturdays and Sundays, because the stock market is then closed. Thus, we decide to use the option of imputing the weekend values with the preceding Fridays' values. Financial metrics which might be predictive of a short term stock return are the traded volume of the stock and its volatility, as explained in Table 3.1. These are indicators commonly used in trading, which is buying and selling stocks on very short time horizons (Zhou et al. 2018), (Oliveira et al. 2017). Additional metrics could be collected in the future research in order to increase the prediction accuracy, e.g. moving averages, turnover (the total value of the trade shares) and a more granular data on the prices (e.g. opening price, maximum price, minimum price).

Moreover, we collect additional proprietary Bloomberg metrics as outlined in Table 3.2. We believe that they contain relevant information that can serve as a benchmark for our model. For example, we obtain unique News Heat information, which captures the change in the typical publishing and readership activity. It is measured over the last 32 hour period, and it compares the recent value with the last 30 day period. We believe that such data can have a significant predictive power for the stock return problem, because it signifies a certain change. Regardless of the underlying reason for the unusual activity, it should be reflected in the stock price of a respective company (assuming that financial markets are informationally efficient most of the time). Moreover, the inclusion of Bloomberg's data allow us to extend the scope of our study to include the sentiment reflected in news articles in addition to Twitter posts' sentiment.

Table 3.2: Bloomberg's proprietary data

Bloomberg's proprietary metric	Description
Twitter Sentiment Daily Average	The average daily Twitter sentiment for a given stock
Twitter Positive Sentiment Count	The number of positive tweets about a given stock
Twitter Negative Sentiment Count	The number of negative tweets about a given stock
Twitter Publication Count	The total number of tweets about a given stock
News Sentiment Daily Average	The average daily news sentiment for a given stock
News Publications Count	The total number of news articles about a given stock
News Heat Publication Daily Average	The amount of unexpected publication activity

3.2.4 Legal and Ethical Considerations

This project was evaluated against the Legal and Ethical Considerations checklist. There are three aspects of this research which needed to be addressed:

- Project uses software for which there are copyright licensing implications
- The project produces information for which there are data protection, or other legal implications.
- The project involves further processing of previously collected personal data (secondary use).

To comply with the Bloomberg's software licence, the dataset containing the Bloomberg metrics cannot be published. This limitation is clearly stated in the Bloomberg's Terms of Service:

"Except as expressly permitted by Bloomberg in writing you may not copy, reproduce, recompile, decompile, disassemble, reverse engineer, distribute, publish, display, perform, modify, upload to, create derivative works from, transmit, transfer, sell, license, upload, edit post, frame, link, or in any way exploit any part of the Service, except that you may download material from the Service and/or make one print copy for your own personal, noncommercial use, provided that you retain all copyright and other proprietary notices. You may not recirculate, redistribute or publish the analysis and presentation included in the Service without BLP's prior written consent."

In order to address this limitation, we only use the Bloomberg's proprietary data as a benchmark

for our models, and we do not include them in the publishable dataset. The final dataset does include the financial metrics as outlined in Table 3.1, because they are publicly available data which can be obtained from other sources like Yahoo Finance.

Lastly, we address the third concern regarding the processing of user data. For the purposes of this project we downloaded a large collection of tweets with metadata that included personal information such as the authors' username and their geolocation. In order to comply with personal data protection rules, we decided to retain only tweets' text and date of creation, as these two pieces of information were essential to conduct this research. The remaining tags, including tweet id, username and geolocation, were not included into the dataset. In this way the data we process is depersonalised, and cannot be viewed in the context of its authors.

In general, the reuse of Twitter data is permitted by Terms of Service, as well as it is within the Privacy Policy (Ahmed et al. 2017).

3.3 Dataset generation

3.3.1 Filtering tweets

In order to create a structured dataset of about 1 million relevant tweets, we filter nearly 2TB of raw data. Upon collection, the raw data had the folder structure as outlined in Figure 3.1. We obtain the data in a compressed format, and we use the *glob* module to find pathnames matching a specified pattern. The json files with tweets are then decompressed using *shutil* module. As outlined in Figure 3.1, the target files have one minute granularity. One month of raw data contains roughly 43200 json files, each having a size of about 10Mb, amounting to over 400GB in total.

Reading in such a large amount of data at once to a Pandas dataframe is infeasible due to RAM limitations. Thus, we explore the methods for reading the data in batches. One technology we try is a *Dask* library, which uses a familiar *Pandas* interface, but runs on dynamic task schedulers. However, we later find that Python code designed for Pandas library is not directly transferable to Dask. In order to avoid issues arising from incompatibility between the two

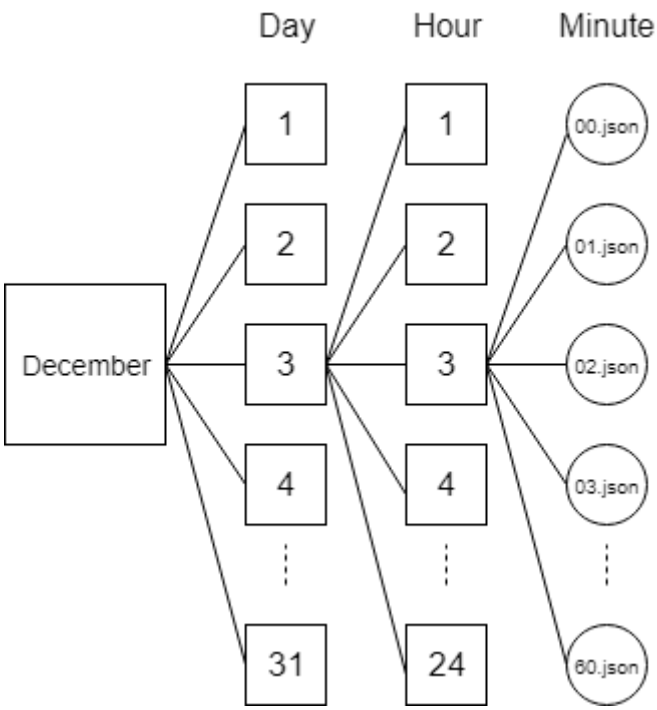

Figure 3.1: Folder structure of the raw data

libraries, we adopt a simpler approach of manually splitting the data into multiple batches, whereby creating daily dataframes. In this process, we filter the tweets by language using a "len" meta-data tag attached to each tweet. Moreover, in the process of creating the dataframes, we discard most of the meta-data tags associated with each tweet, leaving only two field: text and date of creation. This is to comply with the personal data regulations, as well as to reduce the size of the dataset. The procedure is summarised in Algorithm 1.

Algorithm 1 Filter tweets by meta-data tags creating daily batches of data

```
df_tweets
for file in 'F:/Tweets-2018/07/01/*/*.json' do
    temp_df = open(file)
    df_eng =temp_df[temp_df['lang'] == 'en']                    ▷ Keep only English tweets
    temp_df = df_eng['created_at', 'text']      ▷ Keep only tweet's date of creation and text
    df_tweets = concat(df_tweets, temp_df)
```

After the dataset is reduced to contain only English tweets, we perform a keyword search to retain the tweets which mention at least one of the 100 chosen companies. Choosing the appropriate keywords is paramount to reducing noise in the dataset. We propose two sets of keywords:

1. Common company name, e.g. Santander

2. Official Twitter company username, e.g. "@bancosantander"

We believe that investigating company Twitter usernames produces a more relevant dataset. This is because this approach reduces the risk of erroneously including keywords which disguise as a company name. For example, using a keyword "Ford" captures tweets not only about the American automaker, but also about the sexual assault allegation made by Christine Ford against Brett Kavanaugh in July 2018. Capturing only tweets containing "@Ford" username eliminates this noise. However, excluding common company names reduces the dataset almost tenfold. Thus, we decide to include tweets selected by either of the two sets of keywords. We later investigate whether using only the username approach improves the accuracy of the predictive models.

In order to process the vast amount of data efficiently, we design Algorithm 2 which traverses each batch of data only once. Tweets containing no keywords are removed from the batch. After the keyword search is performed on all batches, we join the reduced dataframes to create the final Twitter dataset, which contains 862,231 tweets.

Algorithm 2 Filter tweets by keywords

```
1: stocks                                   ▷ A list of 100 companies' names/usernames
2: tweetList                                ▷ A list of all tweets
3: keywords = [ ]
4: for tweet in tweetList do
5:     findKeywords(tweet, keywods)
6: procedure FINDKEYWORDS(tweet,keywods)
7:     tokenized = tokenize(tweet)
8:     for stock in stocks do
9:         if stock in tokenized then
10:            keywords.append(stock)
11:            return keywords
12:    keywords.append('NaN')
13:    return keywords
```

3.3.2 Merging Twitter and Bloomberg Data

The two datasets need to be merged together on common indices. Firstly, we look at a tweet's keyword and creation date. Next, we look up the keyword in the dictionary to find the column index that the given company has in the Bloomberg dataset. Having located the relevant column, we search a time series of values to find the one relating to the tweet's date of creation. In order for the dates in both dataset to match, they are turned from strings into *datatime* objects, and their formats are normalised. The process is repeated for all financial metrics and Bloomberg's proprietary metrics.

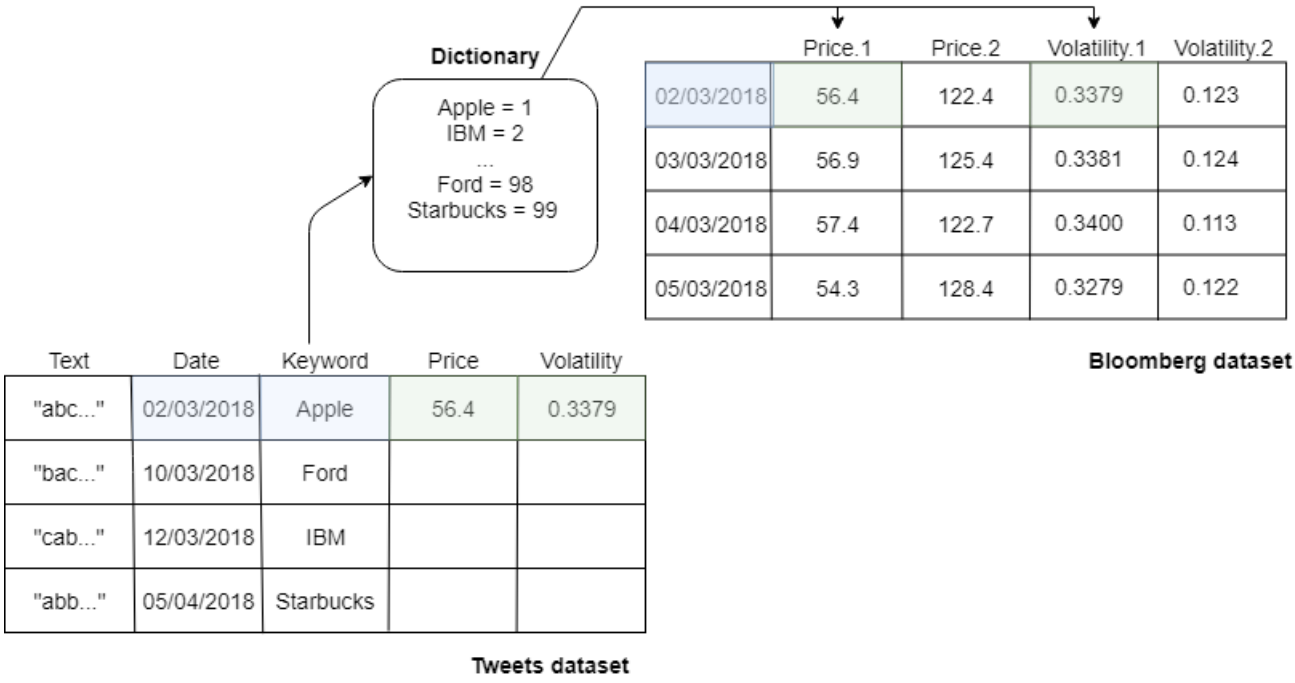

Figure 3.2: Merging Twitter and Bloomberg data

3.3.3 Grouping Independent Events

The resulting dataset is not a typical time series, as there is no guarantee that each company is mentioned in tweets at least once a day. Due to this fragmented nature of the data, we assume that imputing missing value would introduce too much noise. Instead, we make an assumption that every day for any company is an independent event. In the context of the sentiment analysis, we believe that this assumption is likely to hold true. For example, a swathe of angry

customers tweeting about their bad experience with a given company on a given day might have an influence on the company's next day stock price. However, this does not mean that the company's fundamentals have changed. Thus, it is likely due to chance that these customers felt certain way about a given company on a given day. Tweets published next day mentioning the same company might be referring to something unrelated to the past.

We create a group for every independent event. An independent event is formed of all tweets relating to a given company on a given day. We eliminate groups that contain less than 10 tweets. In total, we obtain 5352 groups from the initial 862,231 tweets. Table 3.3 illustrates this methodology and resulting example groups.

Table 3.3: Example groups of independent events

Date	Keyword	Group ID
02/03/2018	Apple	0
02/03/2018	Apple	0
02/03/2018	Apple	0
05/03/2018	Apple	1
02/03/2018	IBM	2
02/03/2018	IBM	2
05/03/2018	IBM	3
07/03/2018	Starbucks	4

It follows that we treat all tweets belonging to a given group as one sample. We elaborate on how we achieve the feature grouping in the next chapter.

3.3.4 Attaching labels

We decide to attach four labels to our data: one-, two-, three-, and seven-day stock returns. A stock return is a percentage difference between a company's stock price on the tweet's creation date and the future price. Selecting the aforementioned time lags for generating labels is in alignment with commonly investigated lags in literature (Bollen et al. n.d.), (Li et al. 2014). We are interested in predicting the direction of the stock price rather than absolute magnitude. Thus, we transform the returns into 3 categories: negative return (below 0%), no change (0%) and positive return (above 0%). The obtained dataset from Bloomberg imputes weekend price values with Friday closing prices. Thus, there are a lot of "no change" labels just due to the

fact that a tweet was published on Friday. In order to eliminate this noise, we exclude "no change" label. Thus, the resulting dataset is ready for a binary classification task.

3.4 Conclusion

We search for a suitable dataset that could allows us to answer the posed research questions. Due to the lack of availability of such a dataset, we collect a few terabytes of raw Twitter data, and filter it to generate our own dataset. The dataset contains tweets that mentions at least one of the companies from the Financial Times' Top 100 Global Brands 2019 ranking, in total 862,231 samples. We attach one-, two-, three-, and seven-day return labels to the dataset, and release for the NLP community to use.

Chapter 4

Feature and Polarity Extraction

4.1 Introduction

The data obtained from social media posts contains a lot of noise - it includes hashtags, URLs, made-up word, and emojis. Thus, we first clean the text, as described in Chapter 4.2. However, textual documents cannot be fed directly to the machine learning algorithms. Most of them expect numerical feature vectors with a fixed size, rather than a sequence of symbols with variable length. For that reason we explore multiple vectorization techniques: Count Vectorizer, Tf–idf Vectorizer and Hashing Vectorizer. In addition to these strategies, which are collectively known as Bag of Words or "Bag of n-grams" representations, we also explore the state-of-the-art feature extraction techniques called BERT and fastText. The former method captures contextual information about words, the latter is able to build vectors even for made-up, misspelled or concatenated words. In addition to describing the process of vectorising the text with each of these methods, which we describe in Chapter 4.3, we also outline how we obtain polarity labels for each tweet with TextBlob and LSTM.

4.2 Text preprocessing

Given the unstructured nature of social media posts, tweets may contain characters and words that are not meaningful. Thus, text normalisation is a necessary step in order to prepare the text for further processing. In summary, we performed the following normalisation operations:

1. **Removing URLs** - tweets often contain links. They are not meaningful per se, so we remove them.

2. **Removing mentions** - Users often relate their tweet to another user by means of mentions, e.g. "@sophie1996". We decided to remove mentions, because names of user profiles are not necessarily meaningful.

3. **Removing RT - Retweet reserved word** - Tweets that are re-published by users other than the tweet's author are called retweets, and they begin with a "RT" reserved word. We remove this reserved word, because it carries no sentiment meaning.

4. **Removing hashtags** - Hashtags often consist of a out of dictionary words, so we decide to remove them.

5. **Removing keywords** - We remove company keywords from the tweets.

6. **Changing to lower case** - In order for one word in both lower case and capitalised versions to be considered as one, all words are made lower case.

7. **Tokenising** - In order for words which are followed or preceded by punctuation marks, e.g. "apple?", and "apple," to be considered the same as the word not surrounded by any other characters, e.g. "apple", we need to split sentences into tokens.

8. **Removing stop words** - Stop words are function words that carry little lexical meaning, e.g. "their", "he", "however". However, removing all of them is not advisable for setiment analysis, because certain stop words carry a significant sentiment value, e.g. "no". Thus, we decide to create a custom list of stop words which removes typical English stop words, but keeps words with a negative charge.

Tasks 1-4 were achieved with a tweet-preprocessor, which is preprocessing library for tweet data written in Python. The remaining tasks were performed manually.

4.3 Feature Extraction

4.3.1 Hashing Vectorizer

Hashing Vectorizer is a method of converting a collection of text documents into a matrix of a pre-defined length. This technique applies a hash function to the features (words in vocabulary), uses the obtained hash values as feature indices, and updates the resulting vector at those indices. We decide to obtain features using Hashing Vectorizer, because of its low memory requirements. This is coming from the fact that Hashing Vectorizer can build document representation for all documents in one single pass over the data. However, Hashing Vectorizer has certain disadvantages. Firstly, it is not possible to compute the inverse of the features back to the words. It makes this method impractical for inspecting the impact of particular words on the model. Moreover, choosing too small a number of features can results in hash collisions, which decreases the representation accuracy. Thus, we use Hashing Vectorizer as a benchmark feature extractor. We import the Hashing Vectorizer from scikit learn library, and we set the number of features to the total vocabulary size. This choice minimises both the collisions and the number of empty buckets.

We decide not to use Hashing Vectorizer representation on tweets after they are grouped into independent events. This is because the results of the hashing functions might not necessarily be meaningful when summed up or averaged. Thus, we investigate other feature extraction methods for the grouped data, as described in the following sections.

4.3.2 Count Vectorizer

Count Vectorizer is a simple Bag of Words method of turning a text into a numerical representation. Count Vectorizer ignores the relative position of words in the documents, which are

describe solely by the number of word occurrences. The resulting matrix will have NxM shape, where N is the number of tweets in the dataset, and M is the total number of words in the vocabulary (i.e. features). Thus, each document will have many feature values that are zeros, as shown in Figure 4.1, so the resulting matrix is stored in the sparse format.

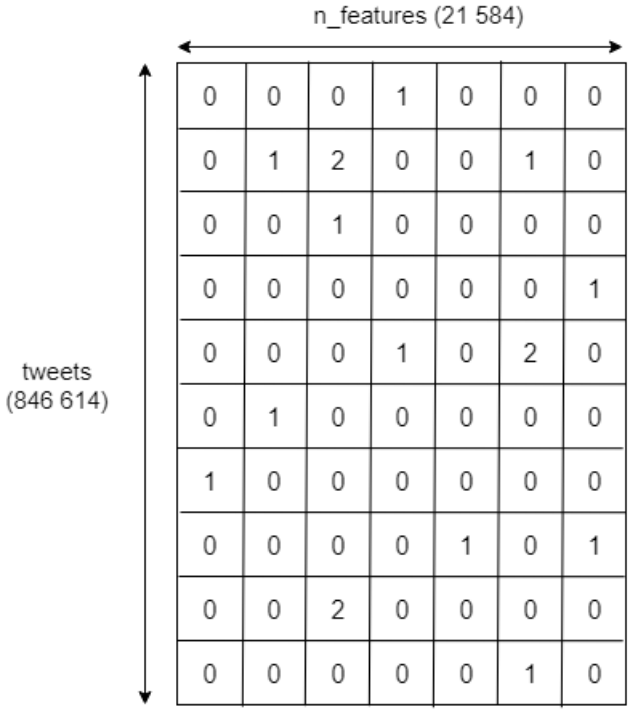

Figure 4.1: Illustration of the Count Vectorizer representation

In order to obtain a Count Vectorizer representation for each group of tweets, we sum the occurrences of each word across all tweets in the group. Effectively, it produces an equivalent result to treating multiple tweets as one input sentence.

4.3.3 Tf-idf Vectorizer

Tf–idf stands for Term Frequency Inverse Document Frequency. Term frequency is the number of times a term occurs in a given document, while the document frequency is the number of times a term occurs in the entire document. We obtain Tf-idf by multiplying term frequency with the inverse document frequency. If the word is used in many documents (tweets), then Tf-idf will decrease. This technique prevents frequent words from being over-represented as a feature, which would then carry little meaning and possibly mask other less frequent, yet more

interesting terms.

4.3.4 FastText

FastText is an extension to the word2vec model, which was proposed by Facebook in 2016. The phenomenon of the word2vec was that it detects similarities between words mathematically, and projects them to a multi-dimensional vector space. There are two version of the model - the Continuous Bag-of-Words model (CBOW) and the Skip-Gram model. CBOW predicts target words from source context words, while the skip-gram does the inverse and predicts source context-words from the target words. At every iteration word2vec evaluates the errors, and follows an update rule that has some notion of penalizing the model parameters that caused the error. By this backpropagation the word2vec learns the word vectors.
FastText distinguishes itself from word2vec by breaking words into several n-grams (sub-words), instead of feeding entire words into the Neural Network. FastText's advantage is that it can properly represent rare words, since it is likely that some of their n-grams also appears in other words.
We import fastText from gensim library. FastText generates a vector of a pre-specified length for each token in our sentence. However, the number of tokens varies from tweet to tweet, and we need a uniform representation for all tweets. In order to achieve vectors of the same length for each tweet, we use the technique called Sentence Embeddings (Arora et al. 2016). We import fast sentence embedding library which generates our vectors with a length of 100 for each tweet.
In order to obtain the FastText representation for each group of tweets, we sum all the sentence embeddings belonging to one group.

4.3.5 BERT

At the end of 2018, researchers at Google proposed BERT (Bidirectional Encoder Representations from Transformers), a novel way of obtaining features which accounts for the polysemous

nature of words (Devlin et al. 2018). The authors of BERT recognised that in order to obtain meaningful embeddings, it is necessary to view words in the context of the surrounding words. For example, the word "shelf" will have a different meaning in these two sentences: "I put my book on the shelf", "The ice shelf is melting due to global warming". BERT is the first unsupervised, deeply bidirectional system for pre-training NLP. Unsupervised means that BERT was trained using only a plain text corpus (Wikipedia + BookCorpus). BERT represents "shelf" using both its left and right context. BERT's approach is the following: it masks out 15% of the words in the input, runs the entire sequence through a deep bidirectional Transformer encoder, and then predicts the masked words. It was trained on a large model (12-layer to 24-layer Transformer). According to the paper which proposed the new technique, BERT obtains new state-of-the-art results on multiple natural language processing tasks such as natural language inference, paraphrasing, named entity recognition or question answering.
We use the pre-trained BERT-Large, Uncased (Original) model in order to construct word embeddings for our tweets. Firstly, we import a pytorch interface for BERT by Hugging Face. Next, we use the interface's functionality to further pre-process the data in order to change it to the format that BERT expects. For obtaning BERT embeddings we use BERT-specific tokenizer instead of the TweetTokenizer that we used for other feature extraction techniques. The Bert Tokenizer has three main steps:

1. **Text normalization -** Converts all whitespace characters to spaces, lowercases the input and strips out accent markers.

2. **Punctuation splitting -** Add whitespace around all punctuation characters.

3. **WordPiece tokenization -** Apply whitespace tokenization to the output of the above procedure, and apply WordPiece tokenization to each token separately.

WordPiece tokenization greedily creates a fixed-size vocabulary of individual characters, sub-words, and words that best fits our language data. Thus, WordPiece tokenizer first checks if the whole word is in the vocabulary. If not, it breaks it down into the largest possible sub-words contained in the vocabulary. If the word cannot be composed of the vocabulary sub-words,

WordPiece decomposes the word into individual characters.

After the tokenized text is fed into the model, BERT produces the 4-dimensional objects in the following order:

1. Number of layers (predefined to 12)
2. Number of batches (1)
3. Number of tokens in a sentence
4. Number of features (predefined to 768)

However, we want to obtain a single vector representation of one sentence. Thus, it is necessary to combine some of the hidden layer vectors. The authors of BERT suggests six solutions, which are visualised in Figure 4.2 along with their performance as cited by the original paper (Devlin et al. 2018).

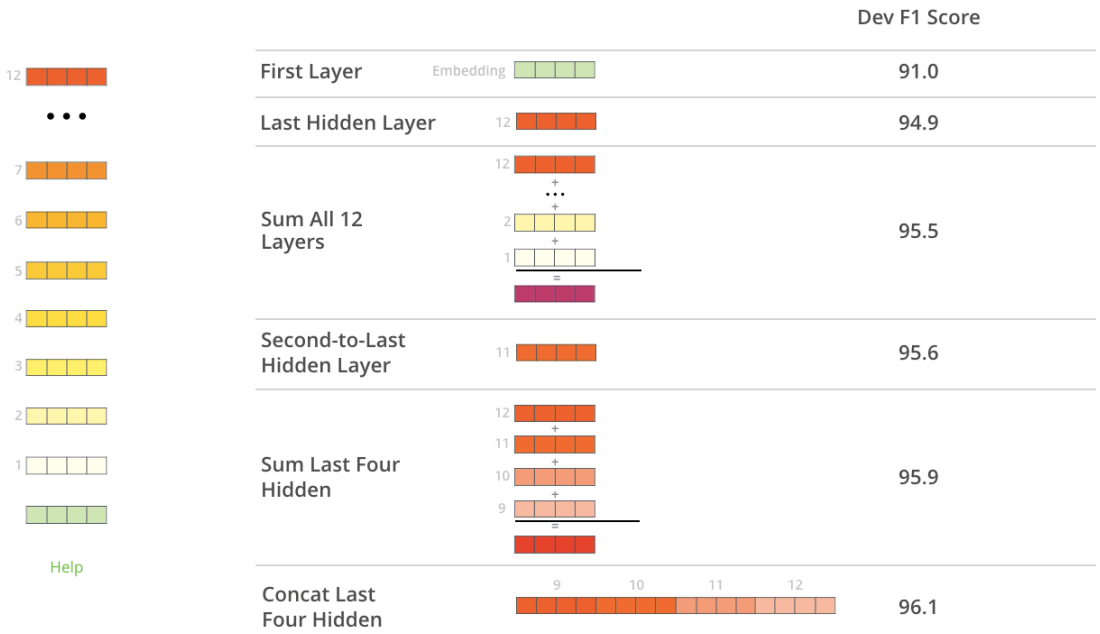

Figure 4.2: Performance of BERT feature-based approach, adopted from Alammar (2019)

The best performing method is to concatenate the token representations from the top four hidden layers. In our algorithm, we choose to use second-to-last hidden layer, which is only 0.5 F1 behind the best-performing method, but it is simpler to implement.

Following the methodology applied to Count Vectorizer and FastText, we sum BERT vectors within each group in order to obtain one aggregate vector for each group.

4.4 Polarity Extraction

4.4.1 Textblob

TextBlob is a Python library for processing textual data, and it includes an API for sentiment analysis. TextBlob uses a simple lexicon approach to generating sentiment for a sentence. The sentiment calculation for one word is done by averaging the polarity of the entries of the word that are found in the lexicon. For example, the word **atrocious** contains 2 entries:

```
<word form="atrocious" sense="provoking horror" polarity="-0.6"/>
<word form="atrocious"  sense="exceptionally bad or displeasing" polarity="-0.8"/>
```

The obtained polarity score is a float within the range [-1.0, 1.0], in this case -0.7. Algorithm 3 is applied to obtain polarity of a tweet (treated as one sentence):

Algorithm 3 Obtaining polarity of a tweet using TextBlob API

```
procedure getPolarityTextBlob(x)
    blob = TextBlob(x)
    return blob.sentiment.polarity
=0
```

In order to obtain the polarity of the entire sentence, TextBlob averages the polarities of the words that are present in that sentence, excluding the words that are not found in the lexicon. For example, the following code prints the corresponding output:

```
sentence = "Atrocious yet beautiful"
print(getPolarityTextBlob("atrocious"))
print(getPolarityTextBlob("yet"))
print(getPolarityTextBlob("beautiful"))
print(getPolarityTextBlob(sentence))
```

```
-0.7
0.0
0.85
0.075
```

Owing to the fact that TextBlob uses a simple average to find the sentiment for a sentence, it is important that the sentences should not contain misspellings. A misspelled word will not be found in a lexicon, and thus it's polarity will not be taken into account for the calculation. In this study, the misspellings were preserved, which might have had an adverse effect on the accuracy of the sentiment scores.

4.4.2 LSTM

Long short-term Memory (LSTM) is an artificial recurrent neural network architecture that can be successfully applied to extracting sentiment from a sentence. We used a TensorFlow based library LSTM-Sentiment-Analysis by O'Reilly and its pre-trained LSTM model in order to obtain polarity labels. The pretrained model was designed to accept 24x250 embeddings, where 24 is batch size and 250 is the maximum sequence length. Each word from the sequence is looked up in the pre-supplied words list, which is accompanied by word vectors pre-trained using GloVe. The matrix contain 400,000 word vectors, each with a dimensionality of 50. The word's index in the words list is appended as a value in the embedding in a position corresponding to that word's position in a sentence. For example, the sentence: "Those impressionistic paintings are true masterpieces" is vectorized as shown in Figure 4.3. In this way, the order of the words in a sentence is preserved. This method has been applied, because it was recommended in the LSTM-Sentiment-Analysis library's tutorial. However, this method produces a largely sparse vectors with a lot of values as zeros. This might be introducing unnecessary noise. Thus, a possible improvement would be to train a LSTM model with vectors of a more suitable dimensionality for the Twitter data.

The polarity of an individual sentence is obtained according to the following algorithm:

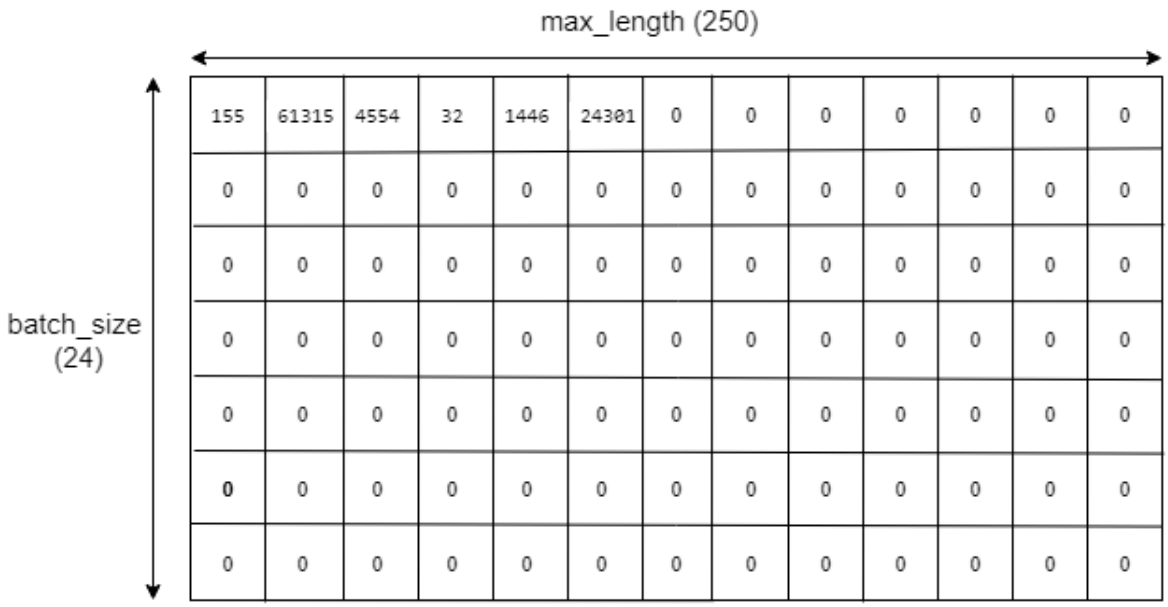

Figure 4.3: LSTM vectors for extracting polarity

Algorithm 4 Obtaining polarity of a tweet using LSTM

```
1: procedure getPolarityLSTM(x)
2:     inputMatrix = getSentenceMatrix(x)
3:     predictedSentiment = getPrediction(inputMatrix)
4:     if predictedSentiment[0] > predictedSentiment[1] then
5:         return 1
6:     else
7:         return -1
```

The *getPrediction* function returns a numpy array with 2 values. Thus, predictedSentiment[0] represents output score for positive sentiment while predictedSentiment[1] represents output score for negative sentiment. We determine the overall sentiment by comparing the positive sentiment with the negative one, and assigning the larger one as an overall sentence sentiment.

4.5 Conclusion

In this Chapter we described several things. Firstly, we outlined our text preprocessing methodology. Secondly, we presented the multiple feature extraction techniques - Count Vectorizer, Tf–idfVectorizer, Hashing Vectorizer, BERT and fastText. We also described the way in which we turn sentence-level vectors into group-level vectors for each feature type. Lastly, we touched on the polarity extraction techniques based on TextBlob and LSTM.

Chapter 5

Data Analysis

5.1 Introduction

In this Chapter we strive to understand the tendencies and themes that occur in our data. Firstly, we perform vocabulary analysis to obtain a general overview of the number of words and the most frequent tokens. Next, we try to understand what distinct themes are present in our data, and whether they show any company-product relationships. We perform this analysis by means of topic modelling (Latent Dirichlet Allocation and Non-Negative Matrix Factorisation) as well as text clustering with K-Means algorithm. We are also interested in visualising the data, so we employ the t-SNE dimensionality reduction technique in order to plot our embeddings. Lastly, we shortly summarise the numerical data that we obtained from Bloomberg.

5.2 Vocabulary analysis

We begin our exploration with analysing the 862,231 collected tweets. We have 142,088 tokens in our vocabulary. Figure 5.1 illustrates the top 20 unigrams before removing stopwords. Figure 5.2 shows how this changes when stop words are removed, but it also reveals an important

problem with the way that tweets are filtered. As we can see, among top 20 unigrams there are words such as "dr", "christine", "blasey", "kavanaugh" and "ford". The reason for their prevalence is that filtering the tweets by the company keyword "Ford", we capture all the tweets relating to Dr. Christine Blasey Ford's sexual assault allegations against Brett Kavanaugh in September 2018. This is confirmed by inspecting a sample of tweets in APPENDIX. It was an important event that had large publicity, and tweets relating to "Ford" constitute 10.9% of the entire dataset. We believe that this is a significant noise. Thus, we remove all tweets with a "Ford" keyword, as we believe that a vast majority of them does not relate to the actual automaker. After removing the "Ford" keywords, we inspect the top 20 unigrams and bigrams as captured in Figures 5.3 and 5.4. We can observe that the most frequent word is "new". This is an expected results, since tweets are often a medium of sharing the most recent information. The same conclusion can be drawn from the inspection of the bigrams, which are dominated by presence of "posted new" and "new video".

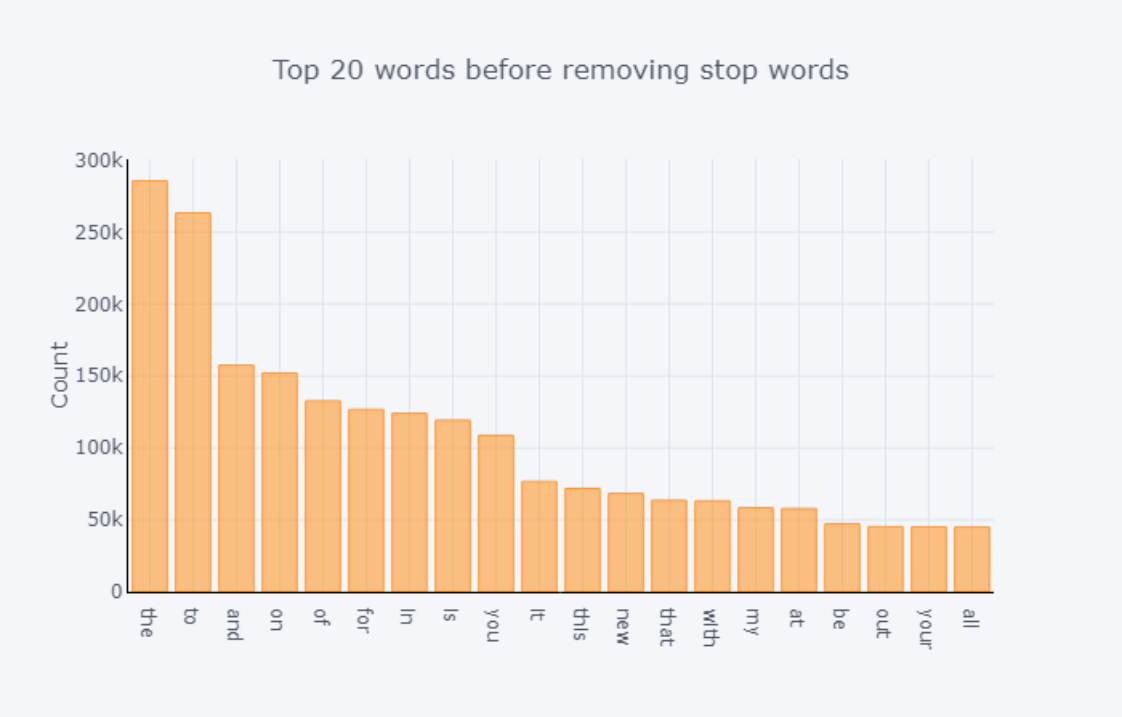

Figure 5.1: Top 20 unigrams before removing stopwords

Moreover, we inspect the typical lengths of tweets, as well as the typical number of tokens they include. As we can see from the Figure 5.5, tweets are most typically up to 120 characters long. This is because the Twitter API only returns 128 characters for each tweets for a historical

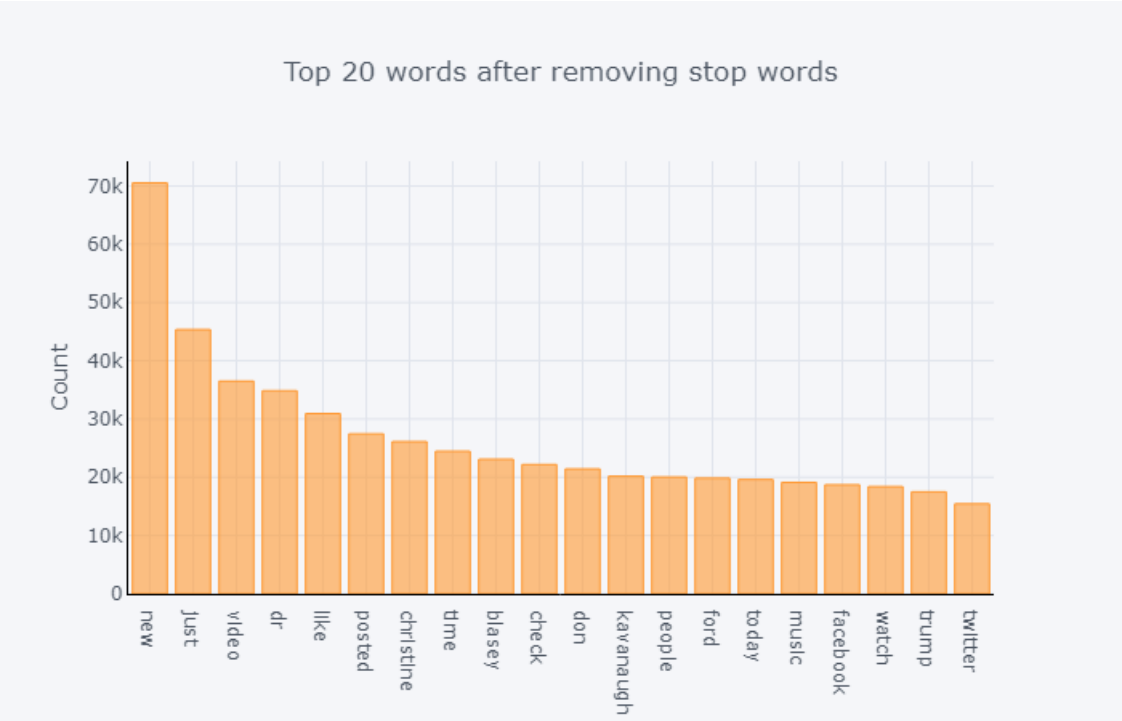

Figure 5.2: Top 20 unigrams after removing stopwords on the dataset containing "Ford" keyword

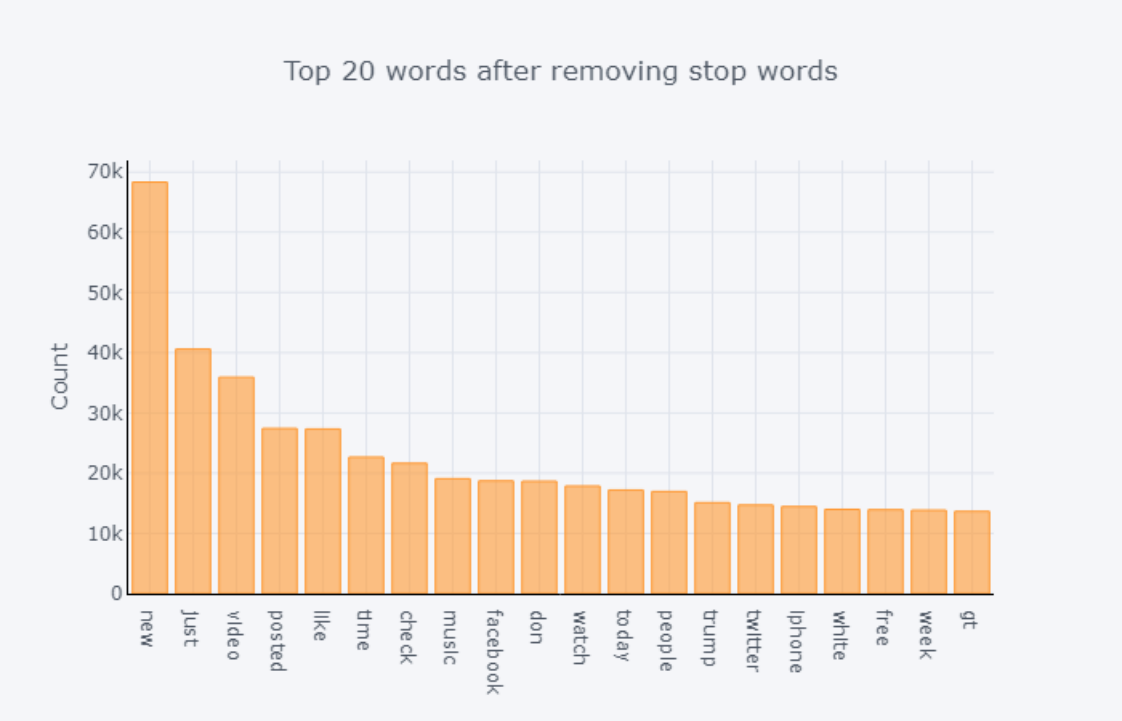

Figure 5.3: Top 20 unigrams after removing stop words on the dataset reduced of 'Ford' keyword

query. The provided length is extended to 256 characters when the query dates back to the last 30 days. The Figure 5.6 reveals that the tweets contain a varying number of words, typically

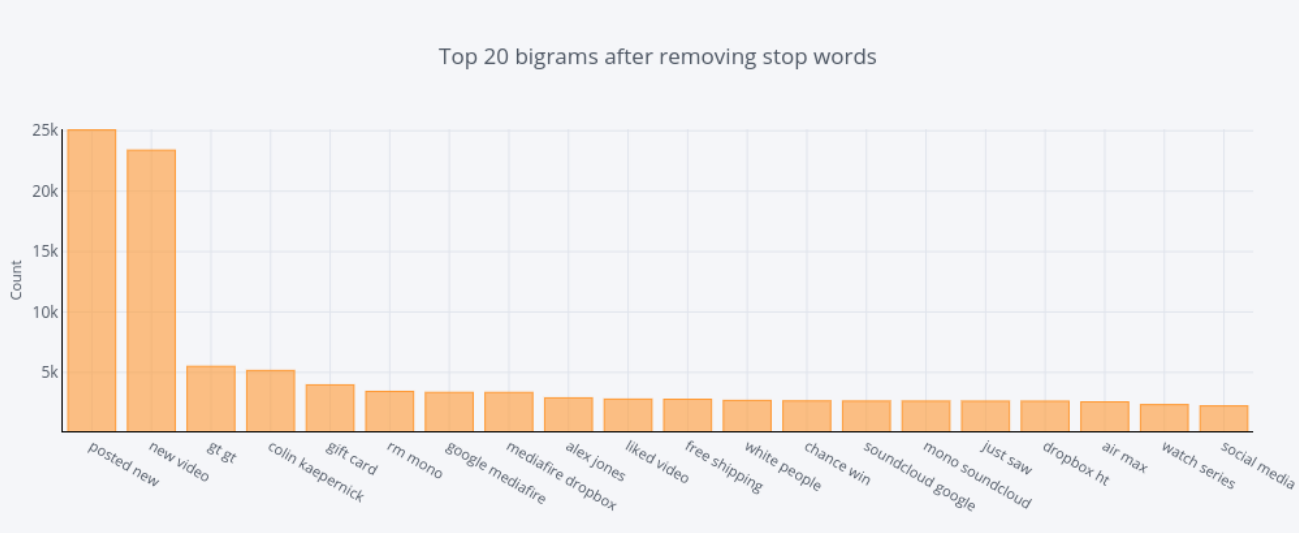

Figure 5.4: Top 20 bigrams after removing stop words and 'Ford' keyword

between 10 and 25.

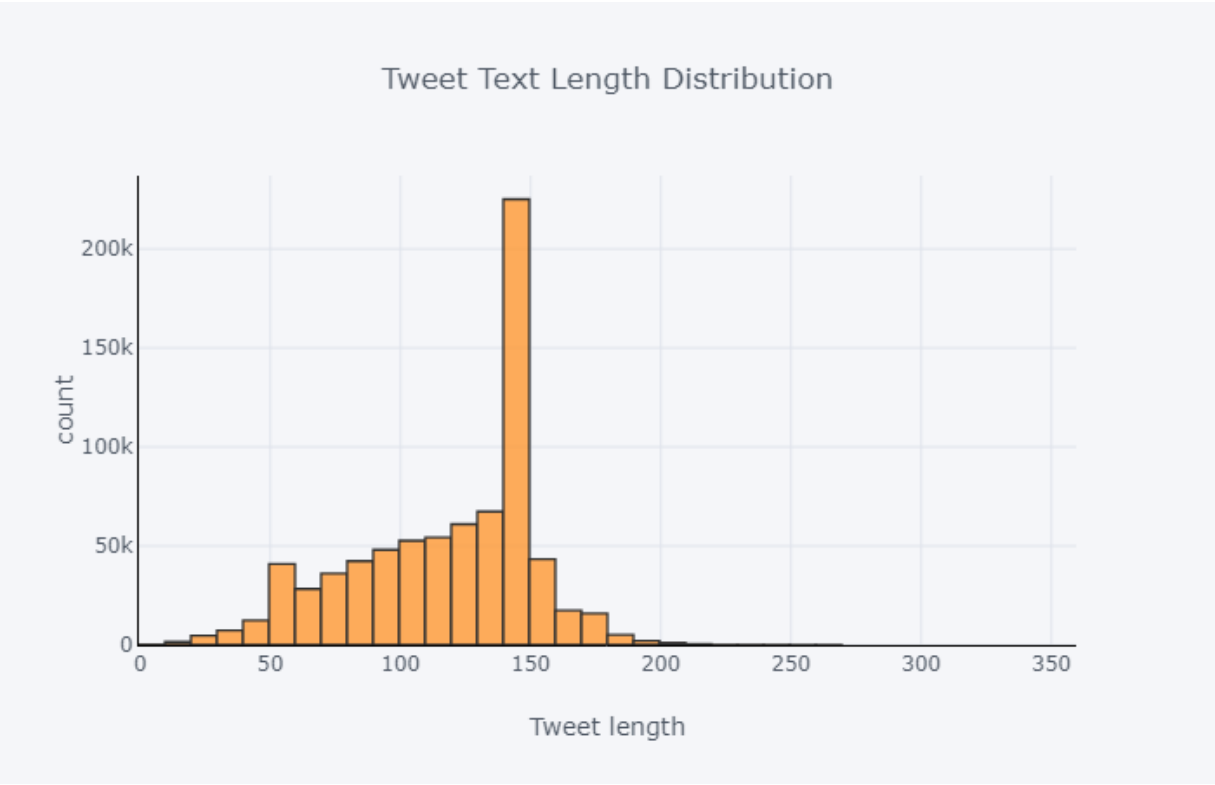

Figure 5.5: Tweet character length distribution

We are also interested in understanding the overall sentiment of the tweets in our dataset. Thus, we inspect the TextBlob polarity scores across our dataset. As evident from Figure 5.7, the dataset is not skewed towards being more positive or more negative. According to the

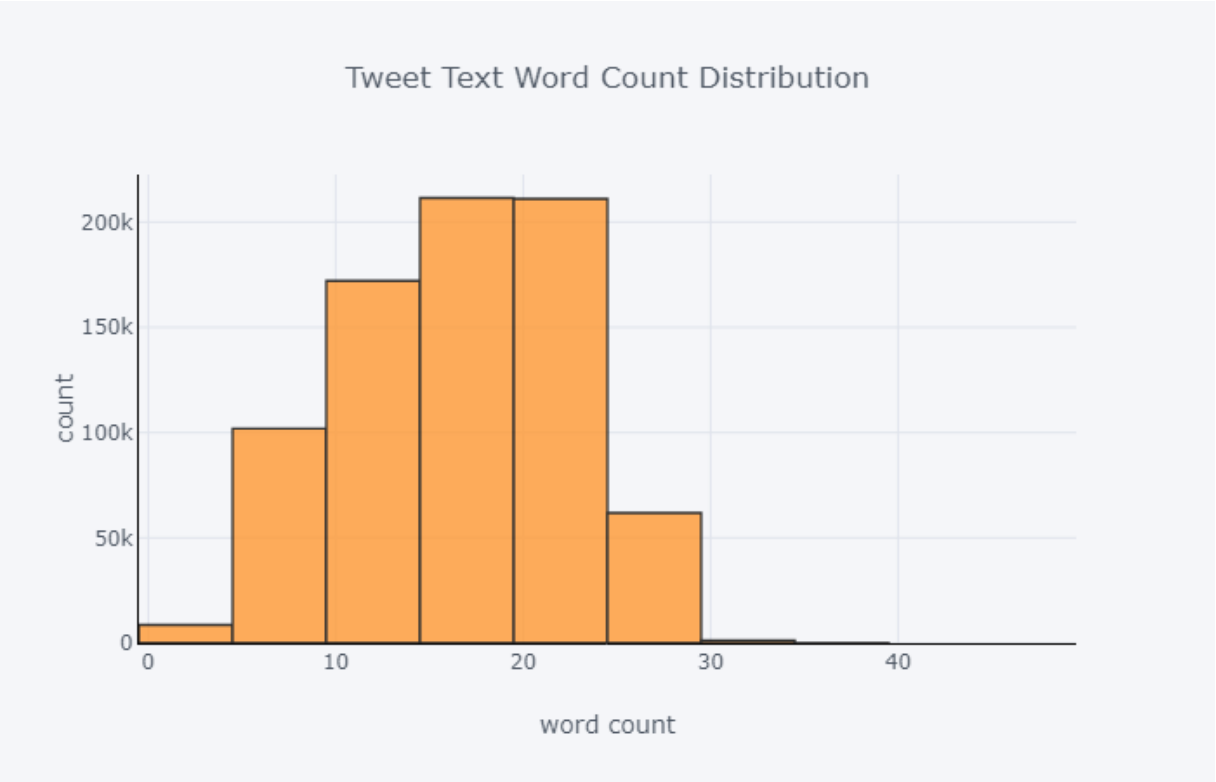

Figure 5.6: Tweet word count distribution

TextBlob sentiment, most tweets are neutral.

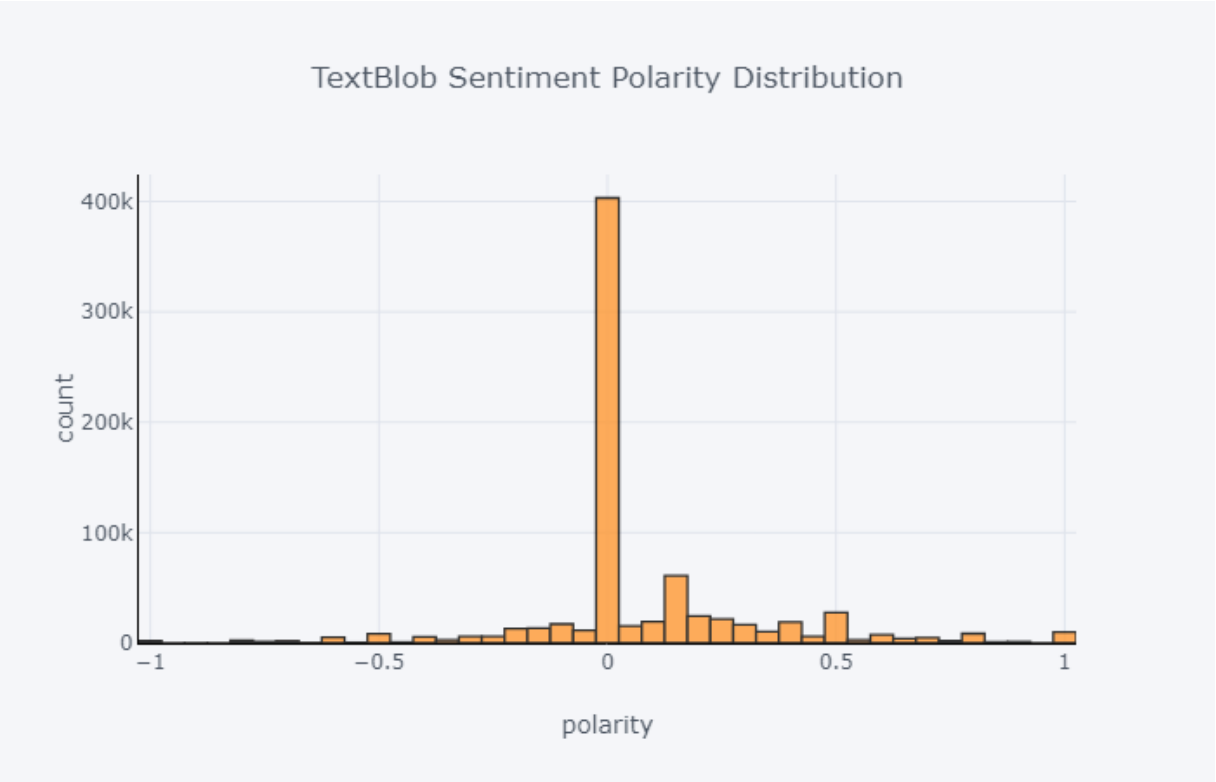

Figure 5.7: The distribution of TextBlob sentiment labels

5.3 Topic Modelling - Dataset Overview

5.3.1 Latent Dirichlet Allocation

Topic modelling is an unsupervised learning method for discovering the abstract "topics" in a collection of documents. One of the techniques to achieve that is called Latent Dirichlet Allocation (LDA). LDA is a generative probabilistic model that assumes each topic is a set of words, and each word in a topic has a associated probability of occurrence in that topic. Moreover, each document is a set of topic probabilities. In this sense, a tweet can be represented by its per-topic distribution (e.g. tweet = 0.3Technology + 0.7Music). Thus, LDA generates a representation for documents in the topic space. In LDA, topic distribution is assumed to have a sparse Dirichlet a prior probability distribition (Girolami et al. n.d.). It captures the intuition that documents cover only a small set of topics and that topics use only a small set of words frequently. We are interested in discovering ten topics that are mentioned in our collection of tweets. We vectorise our data by means of the Count Vectorizer, and we feed them into *LatentDirichletAllocation* from sklearn library. We obtain 10 topics, which can be viewed on an Intertopic Distance Map in Figure 5.8, generated with the *LDAvis* software.

The graph can help us understand two things: how prevalent is each topic and how do the topics relate to each other. Each topic's overall prevalence is encoded using the areas of the circles, where larger circle means greater prevalence. As we can see from Figure 5.8, all the topics have very similar frequency of occurrence in our dataset. It could pose a suspicion that the topics are very similar, however there seems to be little overlap between them on the distance map. We proceed to analysing the compositions of each of the topics, which allows us to understand their meaning. We obtain 10 separate charts for each topic, and we place nine of them in Appendix, showing an example chart for topic 3 in Figure A.11. A pair of overlaid bars represent both the corpus-wide frequency of a given term (blue label) and the topic-specific frequency of the term (red label). By comparing the widths of the red and blue bars for a given term, we can understand whether a term is highly relevant to the selected topic because of its lift (a high ratio of red to blue bard), or its probability (absolute width of red bar). We set

Figure 5.8: Global view of the LDA topic model showing 10 topics in the dataset

the LDAvis-defined relevance measure to 1, which results in the ranking of terms in decreasing order of their topic-specific probability (Sievert & Shirley 2014). For example, in the case of topic 1, we can see that top 10 words in terms of probability of occurrence are: 'new', 'just', 'want', 'got', 'series', 'really', 'netflix', 'right', 'store', 'page'. After inspecting the most relevant terms for each of the ten topics, we suggest 10 broad interpretations of each topic, with top 5 words in brackets:

1. **TV series** (new, just, want got, series)

2. **Music** (music, twitter, people, today, ll)

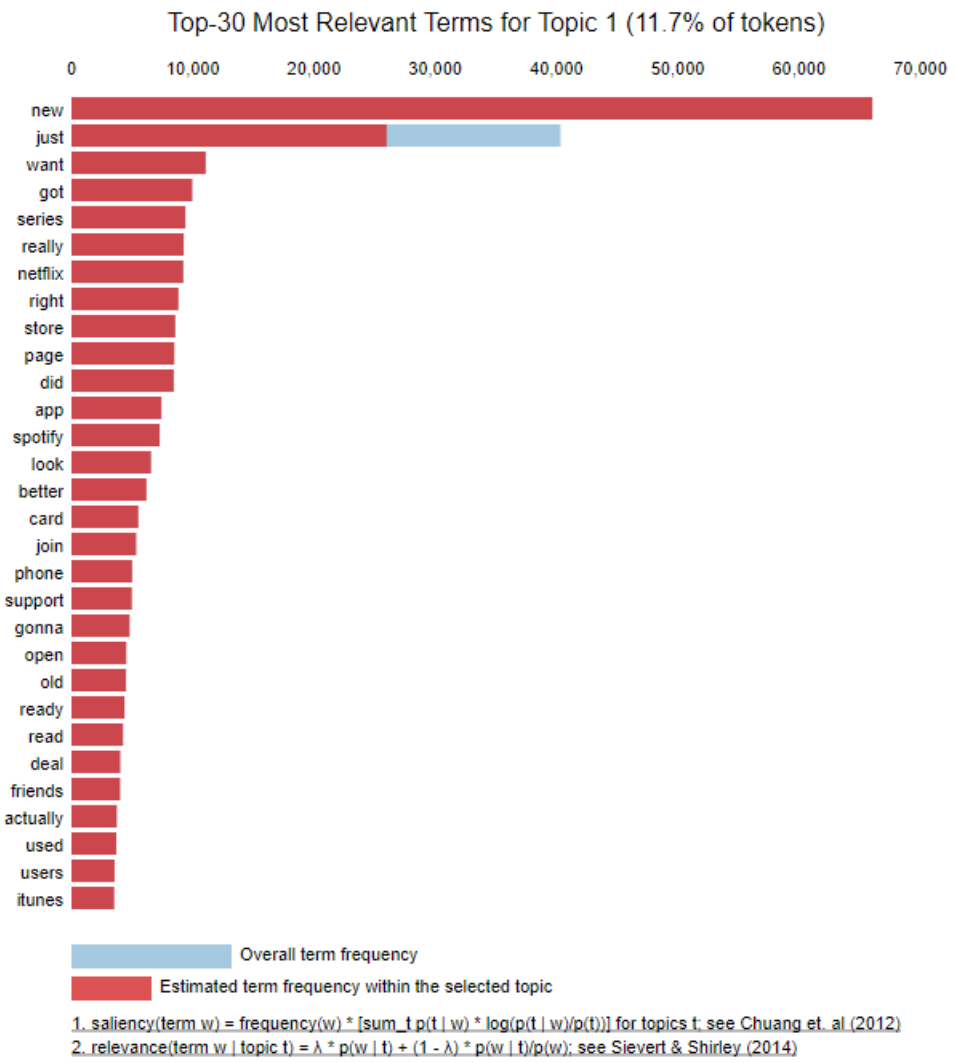

Figure 5.9: Terms that are the most useful for interpreting Topic 1 from LDA

3. **iPhone** (posted, iphone, gt, just, google)
4. **Videos** (video, watch, free, love, year)
5. **Trump** (facebook, trumpt, rt, breaking, face)
6. **Time** (time, like, don, https, think)
7. **Sport** (week, white, kaepernick, day, news)
8. **Shopping** (buy, best, check, stop, ass)
9. **Nike** (know, black, man, nike, air)
10. Noise (need, says, fucking, update, max)

Interpreting the meaning of topics is a difficult task, and the outlined suggestions are open to discussion and re-evaluation. By "Noise" we mean that we could not think of one theme name spanning all the top keywords. We can see that it is rather difficult to find any topics that would relate companies to their products. The only exception is Nike, which appears frequently with the keyword "air", which must refer to Nike Air shoes.

5.3.2 Non-Negative Matrix Factorisation

In order to confirm our findings from LDA model, we explore another method for topic modelling, namely Non-Negative matrix factorisation. While LDA and NMF have differing mathematical underpinning, both algorithms are able to return sets of words that constitute topics. NMF is based on linear algebra - it decomposes the document term matrix into two matrices, W (document to topic matrix) and H (a word to topic matrix) that when multiplied together reproduce the bag of words matrix with the lowest error. NMF achieves this task by optimising the distance d between X and the matrix product WxH. Matrices are constrained to be non-negative, so that components can be interpreted in a meaningful manner. We build our document term matrix with Tf-idf, and feed in into the NMF algorithm from sklearn library. We obtain the following 10 topics:

1. **Brand** (brand,york,added,playlist,ad)
2. Noise (look,win,white,need,want)
3. **Ads** (free,page,listing,air,white)
4. **Trump** (trump,playlist,download,ht,soundcloud)
5. Noise (fucking,dead,day,got,amazon)
6. **IPhone** (iphones,welcome,gb,apple,xr)
7. **College** (got,today,say,college,cutting)
8. **Music** (stream,album,available,play,love)

9. **Social Media** (banned,news,join,live,instagram)

10. **Photos/Sport** (album,camera,sport,gb,online)

The inferred topics overlap with the results obtained from LDA, e.g. "Trump", "Music", "Sport", "IPhone". Moreover, we can see that iPhone appears frequently with Apple. This is another piece of information about the company-product relationships in our dataset.

5.4 Text clustering with K-Means

While topic modelling aims at discovering the latent topics that generated the documents of the corpus, clustering methods' purpose it to partition data into coherent groups. The distance between the samples, which are represented by e.g. tf-idf features, is central for clustering methods. We focus on the popular K-Means algorithm. K-Means performs the clustering in the following steps. Firstly, it chooses the initial centroids (we specify their number to be ten). After this initialisation, K-Means assigns each sample to the closest centroid. The algorithm then updates their centroids, by taking the mean value of all of the samples assigned to each previous centroid. The algorithm calculates the difference between these new centroids and old centroids, and the process repeats until the difference falls below a specified threshold. We obtain the following K-Means clusters for using Tf-idf features:

1. Noise (headphones, dead, look, new, firebrand)

2. Noise (rothys, rothy, loafer, ballet, comfort)

3. Noise (showing, feel, dm, wants, link)

4. **Videos** (posted, new, video, photo, firecracker)

5. Noise (trumping, values, episode, size, fired)

6. **Shopping** (deals, great, store, amazon, yes)

7. **Music** (just, new, time, music, check)
8. Noise (like, don, looks, rt, just)
9. **Videos** (posted, video, new, firecable)
10. Noise (diba, feu, account, usi, discovered)

As we can see, the most frequent words in six out of ten clusters are difficult to interpret as a theme. Moreover, the "Videos" theme seems to be prevalent in two clusters.
We also investigate the resulting the top words in clusters resulting from Count Vectorizer features:

1. **Promotion** (10, generation, giveaway, casio, ends)
2. **Social media** (new, iphone, posted, watch, photo)
3. **Life** (th, life, suite, better, place)
4. Noise (studios, send, day, hulu, good)
5. **Technology** (gt, microsoft, files, rar, sfx)
6. **Shopping** (look, says, update, search, ebay)
7. **Videos** (video, posted, new, art, official)
8. **Facebook** (just, like, time, check, facebook)
9. Noise (cad, turkey, 75, ragu, sponsored)
10. **News** (invited, sky, mnuchin, news, week)

Using Count Vectorizer features seems to improve the clustering. While interpreting clusters is a largely subjective task, it appears that an approximate theme could be attached to eight of them. The themes are largely overlapping with the topics discovered by topic modelling

algorithms, LDA and NMF. The conclusion that can be drawn from comparing the two K-Means findings is that some common words (words which will appear in multiple documents) are helpful in distinguishing between clusters. Thus, when Tf-idf downgrades their importance, the quality of the clustering deteriorates.

Looking at the resulting themes from all topic modelling methods, we do not find any significant company-product relationships in our topics, except for Nike - Nike Air and Apple - iPhone. This suggests that it might be difficult for the algorithm to relate product keywords to the companies' names, and thus the stock return prediction might be a difficult task to perform.

5.5 Topic Modelling Per Company

We decide to approach the problem of modelling company-product relationship from an alternative perspective. Using Tf-idf features and Non-Negative Matrix Factorisation, we decide to find 3 topics for each company in our dataset, and see if the top 3 keywords in each topic are mentioning the companies' products. The analysis shows that the resulting themes within each stock are strongly relatable to the company's products. We report some of the results in Table 5.1. For each company we present selected keywords from the top 3 keywords for each topic. The full output can be found in Appendix.

Table 5.1: Selected keywords from the top 3 keywords found in 3 themes per each company (selected companies only)

Company	Keyword 1	Keyword 2	Keyword 3
Adobe	Photoshop	Illustartor	graphic
Apple	iPhone	MacBook	music
Audi	R8(specific model)	car	new
Aviva	fund	billion	community
Bank of America	account	bank	america
Disney	movie	channel	love
Expedia	hotel	night	star
Heineken	lagos	drinking	ad
IBM	watson	cloud	blockchain
McDonald's	burger	fries	man

This analysis reveals that in our dataset companies are associated with their products. The

presence of such tendencies should help machine learning algorithms to use textual features for the stock return prediction with more success.

This analysis can also help us to identify stocks which introduce noise to the dataset. For example, we examine outputs for Next, TMobile and Wells Fargo in Tables 5.2, 5.3 and 5.4 respectively. None of the keywords seem to be referring to those companies' products. The reasons for it are clear in case of Next. The company name is itself a common word, which appears in many contexts, not necessarily clothes shopping. Wells Fargo is a financial institution and it sells complex financial products, which cannot be captured by means of simple words, like "burger" and "fries" for McDonald's. The reasons for the noise in T-Mobile's themes are more unclear. Overall, we can see that deliberate choice of companies at the dataset generation stage, which avoids companies with ubiquitous names, as well as companies that sell opaque products, is paramount to reducing noise.

Table 5.2: Three topics with top 3 keywords for Next

Company	Top 3 Keywords
Topic 1	next,stop,time
Topic 2	year,week,time
Topic 3	get,next,week

Table 5.3: Three topics with top 3 keywords for TMobile

Company	Top 3 Keywords
Topic 1	world, donate, ll
Topic 2	home, every, hurricane
Topic 3	donate, we, tweet

Table 5.4: Three topics with top 3 keywords for Wells Fargo

Company	Top 3 Keywords
Topic 1	tim, inappropriately, ll
Topic 2	using, math, need
Topic 3	inappropriately, tim, fire

5.6 Dimensionality reduction with t-SNE

We perform the dimensionality reduction with a technique called t-Distributed Stochastic Neighbor Embedding (t-SNE). t-SNE finds a probability distribution that dictates the rela-

tionships between various neighbouring points in the original feature space. The Gaussian distribution is employed as the probabilistic measure for this purpose. t-SNE then recreates that probability distribution in a lower dimensional space as closely as possible. The algorithm achieves that by minimising the sum of Kullback-Leibler divergence, which is a measure of how one probability distribution diverges from a second, using a gradient descent method. The advantage of t-SNE over more classical dimensionality reduction tools such as PCA, is that it can capture non-linear dependencies in the data.

In order to understand the effect of grouping of tweets discussed in Section 3.3.3, we perform dimensionality reduction with t-SNE on both ungrouped and grouped dataset. In order to improve the performance of the t-SNE, we first reduce the embedded space to 50 dimensions with TruncatedSVD. After this operation, we proceed to project our data to the two-dimensional space that can be viewed on the plots below.

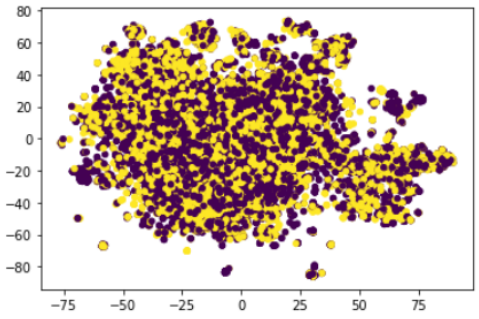

Figure 5.10: 1 Day Return, dimensionality reduction of ungrouped tweets with t-SNE

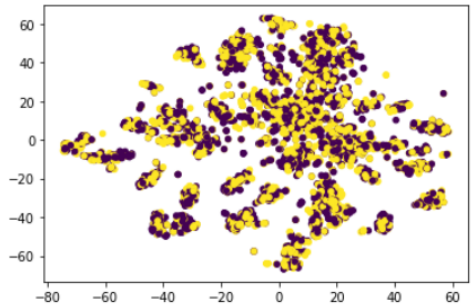

Figure 5.11: 1 Day Return, dimensionality reduction of grouped tweets with t-SNE

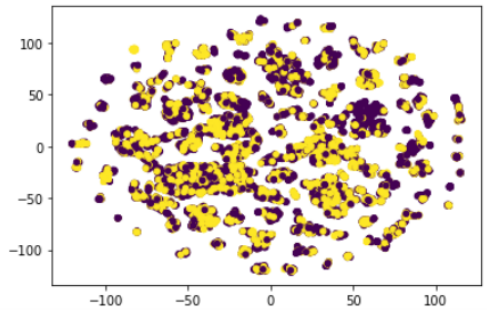

Figure 5.12: 2 Day Return, dimensionality reduction of ungrouped tweets with t-SNE

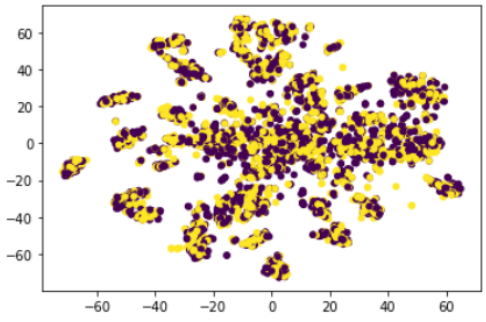

Figure 5.13: 2 Day Return, dimensionality reduction of grouped tweets with t-SNE

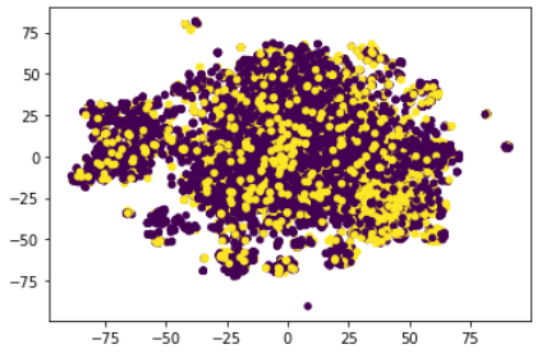

Figure 5.14: 3 Day Return, dimensionality reduction of ungrouped tweets with t-SNE

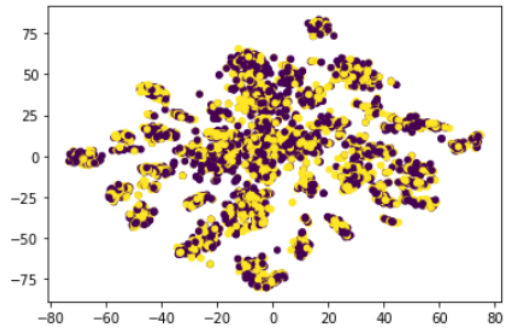

Figure 5.15: 3 Day Return, dimensionality reduction of grouped tweets with t-SNE

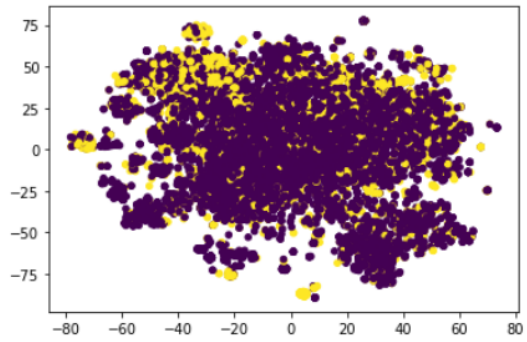

Figure 5.16: 7 Day Return, dimensionality reduction of ungrouped tweets with t-SNE

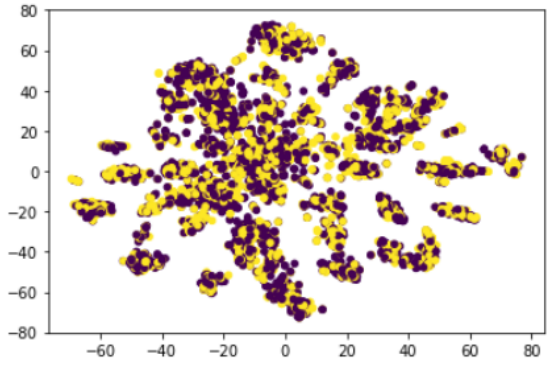

Figure 5.17: 7 Day Return, dimensionality reduction of grouped tweets with t-SNE

We can see that the effect of grouping is the appearance of multiple clusters. Each cluster is one independent event - a collection of tweets about a given company on a given day. Yellow colour signifies a positive return label, and purple means negative return label. The graphs show the inherent difficulty in the capturing discriminative semantics of tweets with positive and negative return labels. Despite grouping the tweets by the proposed methods, both labels are equally well distributed, making the task of classifying the tweets into return categories a difficult problem.

5.7 Numerical data analysis

5.7.1 Descriptive statistics

Table 5.5: Descriptive statistics of the financial data

	1-Return	2-Return	3-Return	7-Return	Volume	Volatility10D	Volatility30D
mean	0.000961	0.001654	0.001414	0.003201	1.312400e+07	25.469685	25.802760
std	0.019171	0.023578	0.027333	0.043026	1.585069e+07	19.411511	13.970805
min	-0.146650	-0.173554	-0.177851	-0.204959	1.000000e+00	0.619000	4.525000
max	0.233973	0.243639	0.243639	0.267113	3.148332e+08	124.137000	87.685000

Table 5.6: Descriptive statistics of the Bloomberg's proprietary news data

	SentimentDAvg	PublicationCount	HeatPubDAvg
mean	0.109402	942.302862	1.253843
std	0.268245	1398.927987	0.913221
min	-0.998600	1.000000	0.000000
max	0.992500	15332.000000	4.000000

Table 5.7: Descriptive statistics of the Bloomberg's proprietary Twitter data

	PublicationCount	SentimentDAvg	PosSentimentCount	NegSentimentCount
mean	1969.431491	-0.003383	97.297756	137.783272
std	2920.721787	0.060415	241.547479	340.340313
min	1.000000	-0.989000	1.000000	1.000000
max	28334.000000	0.921800	5348.000000	3998.000000

The descriptive statistics of the financial data described in Table 5.5 reveals that the seven-day return has the greatest variation of all four labels - it contains the most negative return (-20.4%) and the most positive return (+26.7%) in the dataset. Volume, which is the amount of stock that was traded on a given day, has a very large range in our dataset - from 1 to 314 mln. The mean 10D and 30D volatilities are similar, but according to expectations, the 10D volatility has a larger standard deviation, and the difference between minimum and maximum values is larger.

The descriptive statistics of Bloomberg's proprietary news data indicates a large variability in the news sentiment and stock coverage in the news articles. SentimentDAvg metric, which shows the average daily news sentiment for a stock and has a [-1,1] range, has almost the maximum variability in our dataset, -0.999 to 0.993. PublicationCount, which shows the total

number of news articles about a given stock varies greatly between 1 and 1399. The same is true about HeatPubDAvg, which is the amount of unexpected publication activity, with the minimum and maximum values being at the edges of the [0,4] range.
Analogical metrics for Twitter data in Table 5.7 also show great variability between tweets in the number of publications and the daily sentiment. It is also evident that on average there are more tweets published that carry a negative sentiment about stocks than positive ones.

5.7.2 Linear Regression analysis

We perform an OLS regression to investigate if any of the financial variables and Bloomberg proprietary metrics explain the one-, two-, three- or seven-day returns. To evaluate the overall fit of a linear model, we use the R-squared value. R-squared captures the proportion of variance explained by the model variables. As we can see in Table 5.8, the financial and proprietary Bloomberg metrics explain about 20-30% of the variance in the data. Our task will be to determine if the textual data can help to explain a larger proportion of the variance in returns.

Table 5.8: Overall fit of the OLS model with all financial and proprietary metrics

	1Day Return	2Day Return	3Day Return	7Day Return
R2	0.222	0.232	0.298	0.257

5.7.3 Partial Least Squares Regression analysis

However, we suspect that the variables might suffer from multicollinearity. Thus, we also perform Partial Least Squares, which is designed to handle the case of a large number of correlated independent variables. The idea behind Partial Least Squares is to decompose both the design matrix X and response matrix Y, extracting the latent factors which account for as much of the variation in Y. As input to the model we use the data matrix of predictors, with number rows being equal to number of tweets, and number of columns being equal to the number of all metrics, and we test it against the four labels. We decompose the data by keeping two latent variables, and calculate the R2 for each label.

Table 5.9: Overall fit of the Partial Least Square model with all financial and proprietary metrics

	1Day Return	2Day Return	3Day Return	7Day Return
R2	0.185	0.200	0.246	0.233

As we can see from Table 5.9, the Partial Least Square decomposition does not increase the proportion of the variance in the dependent variable that is predictable from our independent variables.

5.8 Conclusion

The theme analysis performed by both topic modelling and text clustering reveals that the most commonly discussed subject on Twitter are videos, music, iPhone, sport, Trump, and shopping. This is an expected outcome given the nature of social media. After performing the topic modelling separately for each company, we find that there exist significant company-product relationships in our dataset. This suggests that it might be feasible for a machine learning algorithm to relate product keywords to the companies' names for the purposes of the stock return prediction. However, plotting our data with t-SNE revealed the inherent difficulty in classifying tweets into return categories, as the labels seem to be equally distributed.

Chapter 6

Stock Return Prediction & Evaluation

6.1 Introduction

In this Chapter we strive to answer two research questions:

1. Does the semantic information in tweets improve the benchmark accuracy of one-, two-, three-, and seven-day stock return predictions?

2. Do multiple kinds of features fused together in the multi-view learning process improve the accuracy of one-, two-, three-, and seven-day stock return predictions?

Firstly, we describe several machine learning algorithms that we employ in our investigations, namely Support Vector Machine, Gradient boosting, Feedforward Neural Networks, Multi-view learning and Auto-sklearn. We then proceed to designing three benchmarks. Benchmark 1 contains only financial information. Benchmark 2 contains financial information and "counts", e.g. the number of news articles released, or number of tweets released about a given stock. Lastly, Benchmark 3 contains sentiment information from Bloomberg. In order to answer the first research question, we need to compare the Benchmarks 1-2 to the Benchmarks 3, as well as single-view and multi-view models. To answer research question two, we must compare the results of our multi-view model with Benchmark 3, and single-view feature models. After

delineating the benchmarks used in this study, we proceed to designing our multi-view model in section 6.3.2. Next, we proceed to reporting our findings on both full and reduced datasets, which we then evaluate in section 6.5.

6.2 Machine Learning techniques

6.2.1 Support Vector Machines

A Support Vector Machine (SVM) is a classifier that, given labelled training data, outputs an optimal hyperplane assigning new examples into specified label categories. In two dimensional space this hyperplane is a line dividing a plane in two parts corresponding to the classes. SVM is versatile, and allows for using multiple kernel functions. We experiment with multiple kernel functions provided by the sklearn library, notably rbf, poly and sigmoid. We find the best results with rbf and we use this kernel function for all the predictions. We implement our predictions by means of $C - SupportVectorClassification$ algorithm from sklearn library.

6.2.2 Gradient boosting

Boosting is a method of converting weak learners into strong learners.A weak learner is defined as one whose performance is at least slightly better than random chance. In the case of gradient boosting, decision trees are the weak learner. Trees are added one at a time to the model, and existing trees in the model are not changed. A gradient descent procedure is used to minimise the loss when adding new trees. We import $GradientBoostingClassifier$ from sklearn library. We calculate the training accuracy for several learning rates (0.05, 0.1, 0.25, 0.5, 0.75, 1) with a default number (100) of boosting stages and report the best result.

6.2.3 Feedforward Neural Networks

Feedforward Neural Networks are a class of machine learning models that are inspired by biological neurons and their connectionist nature. The building block of any neural network is a neuron. Every input to a neuron is multiplied by a corresponding weight and then summed to obtain a pre-activation value. This is then passed to a non-linear activation function. We use sigmoid as the activation function in the output layer in order to get a probability distribution between 0 and 1 as a result. In our architecture, we include one hidden layer with the number of hidden units that is twice the dimensionality of the input. In this way we ensure that the network has enough capacity to learn the target function. The loss function is the function we are trying to minimise such that when we do so we learn the relationship between the given inputs and the desired outputs. Because we perform binary classification task, we use the binary cross entropy loss function. The neural network minimises loss by adjusting weights in the correct direction. For this, the derivative of the loss function with respect to the weight is used. The network back-propagates the gradients through the network layers, and minimises the loss by iteratively adjusting weights. We use Adaptive Moment Estimation (Adam) optimiser to perform the gradient descent, because we find that it provides better convergence than classical stochastic gradient descent algorithm. We design the neural network with the help of *pyTorch* library.

6.2.4 Multi-view learning

The various feature extraction methods that we use capture different semantic information contained in a text. For example, CountVectorizer captures the information about the occurence of a given word in a document, but it does not capture the contextual information. This can be captured by means of other algorithms, e.g. BERT. On the other hand, FastText vectors can help us retain the information contained in rare words by making use of character level information.

We can see that there are multiple ways in which we can represent the same textual document. Thus, we believe that we can employ the method known as multi-view learning to improve the

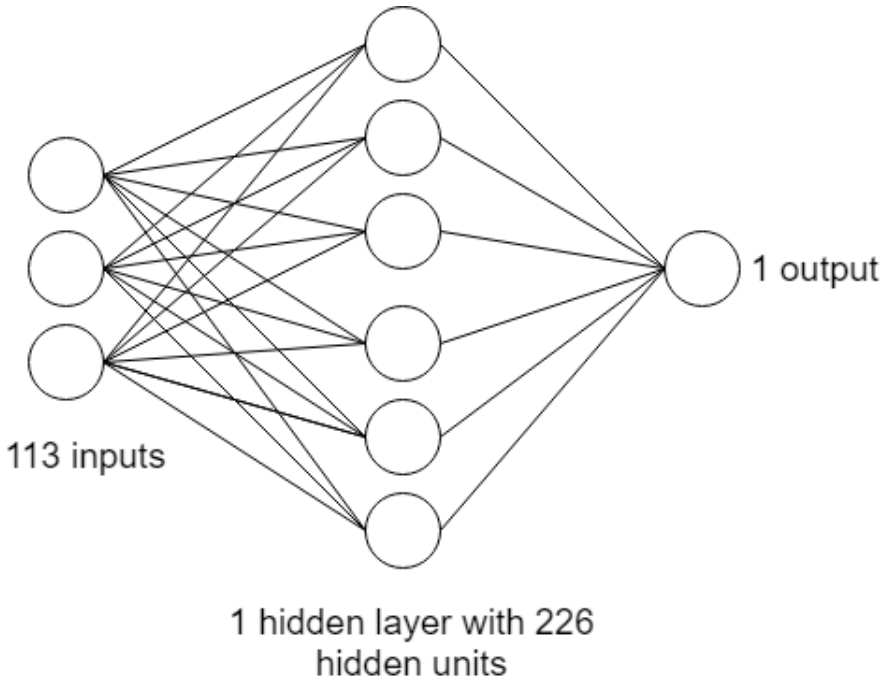

Figure 6.1: The architecture of the Feedforward Neural Network for binary classfication

predictive power of our models. Vectors from differet feature extraction methods are different "views" of the same object - the tweet. In multi-view learning, we project the two vectors into lower dimensional vector space, such that the correlation between these two vectors is maximised in the new feature space. Lastly, we concatenate the resulting vectors. In order to perform this analysis we use CCA *CanonicalCorrelationAnalysis* algorithm from sklearn library. We select the number of dimensions to 50, which results in the 100-dimensional final vector which we later use for return prediction.

6.2.5 Auto-sklearn

Auto-sklearn is an automated machine learning toolkit that uses 15 classifiers, 14 feature preprocessing methods, and 4 data preprocessing methods from scikit-learn library. Auto-sklearn improves on existing automated machine learning methods by automatically taking into account past performance on similar datasets, and by constructing ensembles from the models evaluated during the optimisation (Feurer et al. 2015*a*). We run the auto-sklearn for 180 seconds to obtain the predictive accuracy from the combination of the best performing algorithm and the best configuration of hyperparamteres. We limit the time for algorithm to find the best

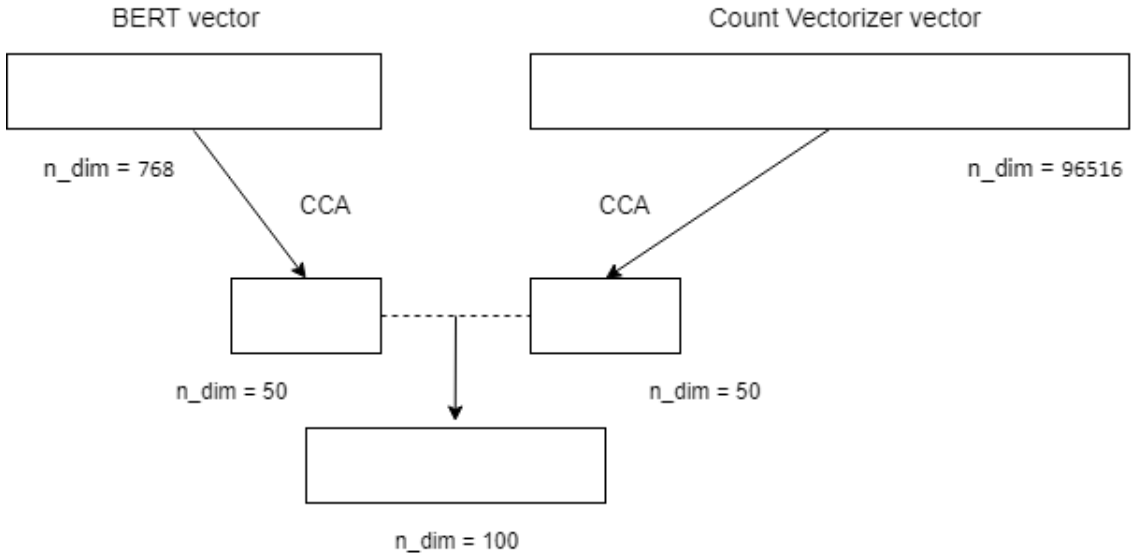

Figure 6.2: The multi-view learning illustration

model just to 3 minutes for practical reasons. We have got 2 dataset versions, 4 labels, and 9 input combinations to test, so allowing the default 1 hour for each prediction would take 72 hours just for auto-sklearn.

6.3 Benchmarks and Model design

6.3.1 Benchmarks

We establish three following benchmarks which include a combination of financial data and proprietary metrics as inputs:

1. **Financial Information**: Trading Volume + 10 Day Volatility + 30 Day Volatility + Last Price
2. **Financial Information + Counts**: Trading Volume + 10 Day Volatility + 30 Day Volatility + Last Price + News Heat Publication Daily Average + News Publication Count + Twitter Publication Count
3. **Financial Information + Counts + Sentiment Information**: Trading Volume + 10 Day Volatility + 30 Day Volatility + Last Price + News Heat Publication Daily Average +

News Publication Count + News Sentiment Daily Average + Twitter Publication Count + Twitter Positive Sentiment Count + Twitter Negative Sentiment Count + Twitter Sentiment Daily Average

The output in each case is one-, two-, three- and seven-day stock return. The division into three different benchmarks helps us to understand what kinds of financial, news and social media signals prompt investors to buy or sell stocks. The first benchmark explain if purely financial metrics are predictive of stock return. The second benchmark includes the "counts' data - the number of unexpected news publications about a given stock, as well as the number of tweets about a given stock. This benchmark does not contain any information about the content of these news articles and tweets; it is purely their quantity that is considered. Lastly, we include the sentiment information in the third benchmark. This last benchmark, which uses Bloomberg's textual analysis metrics, is particularly useful for establishing whether our textual information analysis brings useful insights, helping to predict stock returns with larger accuracy.

6.3.2 Model

We design a model which takes into account the following information:

1. **Semantic information** captured by CountVectorizer, fastText and BERT features
2. **All Benchmark Metrics**
3. **Polarity information** captured by TextBlob and LSTM

Due to the fact that the CCA algorithm considers a maximum of two views simultaneously, we test three combinations of features: CountVectorizer + fastText, CountVectorizer + BERT, fastText + BERT. After obtaining the 100-dimensional vector from multi-view learning, we attach the remaining information from points 2 and 3, forming a 109-dimensional input vector. The model is predicting one-, two-, three-, and seven-day returns.

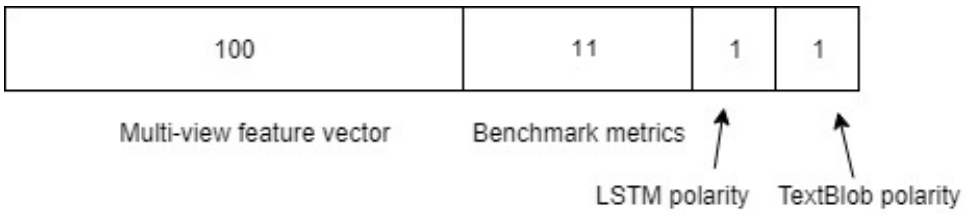

Figure 6.3: The model's input vector

6.4 Results

For our prediction, we test 9 different input combinations (3 benchmarks, 3 single-view and 3 multi-view vectors) with 4 labels (one-, two-, three-, and four-day returns) on 2 different datasets (full and reduced one). For each prediction, we use 80:20 split between training and test datasets.

Firstly, we analyse the results on the full dataset. We begin our investigation with inspecting trends in Benchmark 1-3 results. For day 1 the results for the Benchmarks 1-3 with auto-sklearn are 50%, 55% and 50% accuracy respectively, as can be seen in Table 6.1, which suggests that the prediction is close to random. For day 2, all models report better benchmark results as compared to day 1, e.g. autosklearn for day 2 predicts with 57%, 56% and 55% for Benchmarks 1-3 respectively. This trend continues to 3 day label. Again, the benchmark results show and improvement from day 2 and day 1, with 62%, 58%, and 60% accuracy accuracy for Benchmarks 1-3 respectively. Finally, the prediction for day 7 deteriorates - all benchmarks are predicted with 50% accuracy.

Comparing Benchmark 3, which contains Twitter and news sentiment information, to Benchmarks 1 and 2, which only contain financial information and the number of tweets and articles published on a given day, we can see that sentiment information does not improve the prediction. This is true for all days and models - Benchmark 3 has a systematically lower accuracy than its less information-rich counterparts. The same comparison can be made between Benchmarks 1-2 and any of the features obtained from tweets, both from single view and combined. In this case, for all days except day 7, we also do not note an improvement in prediction accuracy using semantic information from features.

We noted that multi-view features do not improve the Benchmark 1-3 predictions for any of the

days, but they do improve single-view predictions for day 3. Both SVM and auto-sklearn report accuracy of 60-61% for all multi-view vectors, while single view vectors can only be predicted with 52-57% accuracy for day 3.

Finally, we note an interesting anomalous result for day 7, where Count Vectorizer and fastText features with auto-sklearn predict the return with 64% accuracy. This outperforms benchmarks and single-view predictions which stand at 53% for all features with the same model.

Secondly, we analyse the results on the reduced dataset. We observe the same trend in Benchmarks 1-3 as with the full dataset results. The accuracy of benchmark predictions increases up to day 3 with auto-sklearn, after which it drops to random for day 7.

We can observe that this time sentiment information contained in tweets contributes to the improvement in prediction accuracy. This can be deduced by comparing Benchmarks 1-2 with Benchmark 3. For example, for day 2 Benchmark 3 outperforms the other Benchmarks 1-2 by 5 and 3 percentage points respectively using gradient boosting. The same is true for day 3, where Benchmark 3 outperforms the other benchmarks both using gradient boost and auto-sklearn. Similar results are not noted in day 7.

On the reduced dataset, the superiority of the multi-view prediction over single view is more apparent than on the full dataset. We delve into this a little further. For day 2, the best accuracy with for single features is 60% for both fastText and BERT features using gradient boosting, as well as auto-sklearn for fastText. However, combining fastText and Count Vectorizer gives 61% with auto-sklearn. For day 3 the improvement is more remarkable. The combination of Count Vectorizer and BERT features achieves 63%, yet separately they only predict with 57% and 59% accuracy respectively. Similar results are obtained with Support Vector Machine. However, gradient boosting produces conflicting results for this label - the single view features give better prediction accuracy than any of the multi-view vectors. For day 7, again multi-view prediction outperforms single view features with auto-sklearn. The difference is consequential - while fastText and BERT features give 50% and 53% repsectively as singluar inputs, they can predict the 7 day return with 67% accuracy when combined.

Table 6.1: Accuracy of 1 Day Return predictions on the full dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.53	0.51	0.50	0.51
Benchmark 2	0.53	0.52	0.55	0.51
Benchmark 3	0.53	0.52	0.50	0.51
CV	0.53	0.53	0.55	0.52
fastText	0.54	0.54	0.53	0.49
BERT	0.54	0.55	0.53	0.46
Multi-view				
CV + fastText	0.54	0.54	0.53	0.49
CV + BERT	0.55	0.53	0.53	0.51
fastText + BERT	0.54	0.54	0.53	0.50

Table 6.2: Accuracy of 2 Day Return predictions on the full dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.57	0.53	0.57	0.51
Benchmark 2	0.56	0.54	0.56	0.51
Benchmark 3	0.56	0.56	0.55	0.49
CV	0.51	0.54	0.54	0.52
fastText	0.51	0.55	0.53	0.51
BERT	0.53	0.52	0.52	0.52
Multi-view				
CV + fastText	0.56	0.54	0.54	0.52
CV + BERT	0.55	0.55	0.57	0.44
fastText + BERT	0.55	0.54	0.56	0.53

6.5 Evaluation

In this section we try to understand the reasons for the obtained results, as well as their implications for our research questions. Firstly, we asked whether semantic information in tweets improves the benchmark accuracy of return predictions. This question is difficult to answer, because the evidence is mixed. On the full dataset, the Benchmark 3, which contains sentiment information, does not improve the prediction which is made without the sentiment. Moreover, the features obtained from tweets also do not improve the prediction accuracy of the benchmarks, except for day 7, for which we report the combination of Count Vectrorizer and fastText

Table 6.3: Accuracy of 3 Day Return predictions on the full dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	**0.62**	0.52	**0.62**	0.51
Benchmark 2	**0.62**	0.54	0.58	0.52
Benchmark 3	**0.62**	0.56	**0.60**	0.53
CV	0.56	0.54	0.57	0.51
fastText	0.52	0.54	0.55	0.53
BERT	0.52	0.54	0.53	0.51
Multi-view				
CV + fastText	**0.61**	0.54	**0.61**	0.52
CV + BERT	**0.61**	0.56	**0.60**	0.53
fastText + BERT	0.61	0.55	**0.60**	0.52

Table 6.4: Accuracy of 7 Day Return predictions on the full dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.53	0.51	0.50	0.53
Benchmark 2	0.53	0.52	0.50	0.54
Benchmark 3	0.53	0.52	0.50	0.53
CV	0.53	0.55	0.53	0.55
fastText	0.53	0.54	0.53	0.52
BERT	0.53	0.54	0.53	0.55
Multi-view				
CV + fastText	0.53	0.55	**0.64**	0.54
CV + BERT	0.53	0.54	0.54	0.51
fastText + BERT	0.53	0.54	0.55	0.56

features to outperform benchmarks. As we discussed, the results on the reduced dataset are more in favour of the semantic information's predictive power. Benchmark 3 does provide an improvement over Benchmarks 1-2, which evidences the importance of sentiment information. However, semantic information contained in our features does not seem to be more significant for the prediction than benchmark data, except for day 7, where multi-view features do outperform the benchmarks. Overall, it appears that for day 7 the semantic information contained in tweets can explain the returns. However, this is not true for earlier days' returns. It is an interesting result which might suggest that in the timespan of a few days, only the financial information is crucial, but as the time passes, the textual information starts being important too.

Table 6.5: Accuracy of 1 Day Return predictions on the reduced dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.48	0.52	0.51	0.48
Benchmark 2	0.48	0.54	0.50	0.52
Benchmark 3	0.48	0.51	0.53	0.52
CV	0.48	0.55	0.53	0.47
fastText	0.48	0.52	0.49	0.50
BERT	0.48	0.52	0.53	0.43
Multi-view				
CV + fastText	0.48	0.58	0.49	0.50
CV + BERT	0.48	0.52	0.50	0.45
fastText + BERT	0.48	0.51	0.50	0.42

Table 6.6: Accuracy of 2 Day Return predictions on the reduced dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.58	0.59	0.55	0.50
Benchmark 2	0.58	**0.61**	0.58	0.52
Benchmark 3	0.58	**0.64**	0.58	0.52
CV	0.57	0.57	0.56	0.51
fastText	0.57	**0.60**	**0.60**	0.52
BERT	0.57	**0.60**	0.58	0.53
Multi-view				
CV + fastText	0.58	0.59	**0.61**	0.52
CV + BERT	0.58	0.59	0.56	0.54
fastText + BERT	0.58	0.59	0.55	0.52

This can be explained by the nature of algorithmic trading approach to investing, which has become prevalent nowadays. The algorithms, which execute trades in split-seconds, are often built exclusively on numerical data such as trading volume, moving averages, volatility. This information is instantly retrieved from Secondary Information Providers such as Bloomberg, which obtains it first-hand from the stock exchange. Thus, textual data does not have a chance to be absorbed by investors and acted on. Thus, the semantic information contained in tweets is only significant for the prediction with a seven day lag, after human have had a chance to familiarise themselves with it. Other studies also found that there exists a lag between the date of tweets publishing and the time for which it best predicts the returns. According to (Li et al.

Table 6.7: Accuracy of 3 Day Return predictions on the reduced dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	**0.63**	0.51	0.51	0.52
Benchmark 2	**0.63**	0.53	0.59	0.51
Benchmark 3	**0.63**	**0.64**	**0.60**	0.52
CV	**0.61**	**0.60**	0.57	0.48
fastText	**0.60**	**0.60**	0.51	0.46
BERT	0.56	0.58	0.59	0.58
Multi-view				
CV + fastText	**0.62**	0.57	**0.60**	0.52
CV + BERT	**0.61**	0.59	**0.63**	0.58
fastText + BERT	**0.63**	0.56	0.57	0.49

Table 6.8: Accuracy of 7 Day Return predictions on the reduced dataset

	SVM	GBoost	Auto-sklearn	NN
Single-view				
Benchmark 1	0.48	0.52	0.53	0.45
Benchmark 2	0.48	0.54	0.52	0.44
Benchmark 3	0.48	0.51	0.50	0.49
CV	0.48	0.57	0.43	0.48
fastText	0.48	0.53	0.50	0.46
BERT	0.52	0.52	0.53	0.48
Multi-view				
CV + fastText	0.48	0.55	**0.60**	0.42
CV + BERT	0.48	0.50	**0.65**	0.49
fastText + BERT	0.48	0.57	**0.67**	0.50

2017), this time lag is 3 days.

The other research question that we posed concerned the multi-view features, and whether they are able to provide an improved return prediction over the single view features. The evidence is clear that multi-view features do increase the accuracy of return prediction, usually matching the benchmark performance on the reduced dataset. The reasons behind this finding can be understood intuitively - we provide the algorithm with more information (two views) about the same object (tweet). Various feature extraction method capture different kinds of semantics in the text. For example, CountVectorizer captures the frequency of words in a sentence, but not their position. FastText is able to capture character-level information, thus helps us to capture

rare words better. Lastly, BERT provides contextual information - each word is views in the context of its left and right neighbours. Thus combining the features containing different types of semantics in the multi-view learning process results in a more comprehensive understanding of the text by the algorithm.

We also noted an increasing trend in Benchmark 1-3 prediction accuracies between days 1-3. Then, the accuracy falls back to random on day 7 on both datasets. We believe that this finding evidences the widely covered momentum effect in stock market. This phenomenon is characterised by the rising stock prices tending to rise further, and falling prices tending to keep falling. Momentum effect is an anomaly in the stock markets, and arises from the investors' irrational behaviour. Seeing the falling stock prices, they panic and sell their stocks to minimise the loses. On the other hand, seeing the rising prices, they hope to buy before the price peaks to make profit (De Bondt & Thaler 1985). Thus, the financial information available today will be gradually absorbed in the stock price, and it is most predictive of the stock price in 3 days time.

The best results are obtained on the dataset that filters tweets by company usernames, not common names. There is a clear reason for that - by filtering tweets in the more rigorous way, we reduce the noise in the dataset. For example, there might be cases of tweets which get classified as relating to the company "Apple", whereas in reality they only referred to a fruit. Filtering tweets by "@Apple", which is Apple's username on Twitter eliminates this problem.

We can see in Table 6.4 that the best predictive accuracy on the full dataset, 64%, is achieved with the combination of Count Vectorizer and fastText features. However, for the reduced dataset, the best performing combination is fastText and BERT. The reasons for this finding are unclear, but it can be suspected that on the large dataset the frequencies of words carry more meaning, which is better captured by Count Vectorizer. Information in smaller dataset, on the other hand, can be better understood by analysing words' contexts.

We achieve the best accuracy with the auto-sklearn, and it outperforms the remaining classifiers for most input and output combinations. Auto-sklearn tries 15 classifiers, 14 feature preprocessing methods, and 4 data preprocessing methods, and automatically takes into account past performance on similar datasets (Feurer et al. 2015*b*). It is perhaps unsurprising that this

award-winning method outperforms our models with hand-tuned hyperparameters. We note that the classifiers chosen by auto-sklearn that produce the highest accuracy on our dataset are Random Forest and Extra Trees. We make our own predictions with Gradient Boosting, which comes from the same family of machine learning methods, but the automatic hyperparameter tuning done by auto-sklearn results in higher accuracy of the prediction. Despite that, the auto-sklearn's Random Forest and Extra Trees have a tendency to classify the majority of our samples as negative, even when they are not. This might be stemming from the fact that overall more tweets in our dataset have negative sentiment, as reported in Descriptive Statistics in Chapter 5.7.1.

Overall these findings are in accordance with findings reported by (Li et al. 2017) and (Zhang et al. 2018). The first study predicts the direction of returns for a particular company, however, they do not fuse multiple data sources, taking only Twitter sentiment into account. They report the best accuracy, 66.48%, to be with a three-day return label. We marginally outperform this result with our 67% accuracy for the seven-day return. However, the authors of the study also group stocks by industry, and they achieve improved results within IT (76.12%), Finance (70.75%) and Media (73.78%) industries, also for a three-day return. This reveals the importance of dividing stocks into industry categories, and investigating the industry-specific predictability of stock returns. The second study merges two data sources - news articles and social media posts (collected from Guba, another social media site). They quote the performance of their coupled matrix and tensor factorization scheme to be at most 62.50% accurate on the China A-share test dataset. This is outperformed by our 67% accuracy with multi-view learning. However, our findings fall short of the predictions achieved by (Shynkevich et al. 2016). They investigate solely health-care stocks, and they use multiple kernel learning to fuse multiple news sources. They achieve astounding results with 5 Gaussian and 5 polynomial kernels, as well as 5 different news sources - 81.31% accuracy. However, a direct comparison between our results and the findings reported by these studies cannot be made. We quote these findings to illustrate that our results are in a similar range, but since we use a different dataset, the conclusions from their experiments are not strictly comparable to our study.

6.6 Conclusion

In this Chapter we outline the design of the multi-view learning model. It achieves 67% accuracy for a seven day return prediction with auto-sklearn using fastText and BERT features. Our findings are in a similar range as the results quoted in literature. However, by comparing our results with benchmarks it appears that overall textual information does not necessarily improve the stock prediction accuracy for one-, two- and three-day returns. There is some evidence of improvement for the seven-day label after textual features are included. Moreover, our results show that combining multiple kinds of features in the multi-view learning process improves the accuracy of the short-term stock return prediction.

Chapter 7

Conclusion

7.1 Summary of Project Achievements

This project has two main achievements:

1. Creating a labelled dataset which allows for company-level tweet analysis for one-, two-, three-, and seven-day stock return prediction

2. Using that dataset, creating a multi-view learning model with a 67% prediction accuracy, which marginally outperforms literature findings

In order to generate the dataset, we filtered a few terabytes of raw Twitter data. That was a risky endeavour, and we encountered multiple technical problems along the way, notably arising from memory limitations. We overcame the obstacles with designing more efficient algorithms, and using a more sensible partitioning of data into manageable parts. We are proud to announce that our efforts have resulted in success - we have produced a fully labelled dataset for the Natural Language Processing community to use.
In addition to that, we designed a multi-view learning model which achieves 67% accuracy for a seven day return prediction with auto-sklearn using fastText and BERT features. Our model's performance is in a similar range as the results quoted in literature (Zhang et al. 2018), (Li

et al. 2014). We found evidence supporting the hypothesis that combining multiple kinds of features in the multi-view learning process improves the accuracy of the short-term stock return prediction. However, the comparison with benchmarks indicates that the textual information does not improve the stock return prediction overall for the one-, two- and three-day labels. There is some evidence that textual features can help to predict seven-day stock returns. We also acknowledge that the study is bound by a number of limitations that we discuss thoroughly in Chapter 7.3.

7.2 Applications

Our dataset is particularly well-suited for building models for long-term, fundamental investing. Such investing requires a researcher to investigate returns of each individual company separately. In order to understand the applicability of our search for buy-and-hold investors, we discussed the project with the Head of Data Insights at Schroders, a British asset management company. His team is striving to augment investors' decision-making process with machine learning algorithms. The discussion revealed a potential application in detecting the right timing to invest in a stock. During the meeting the attendees pointed out that the decision-making process of an investor might take about one month. Thus, our model can help to choose the right time to make the purchase at the lowest price within that period.

7.3 Limitations

A major source of limitation in this study is the incomplete availability of Twitter data. The collected tweets were only obtainable in the Spritzer form, which captures merely 1% of tweets published on a given day. Any semantics derived from such a proportion of tweets might not reflect the general mood present in the totality of the social posts on a given day.
Another limitation is not taking into account the popularity of the tweets' authors, as measured by number of followers. It is reasonable to assume that users having more followers would have

a larger impact on other people's opinions about a given brand. This in turn could lead to potential increase or decrease in the mentioned company's sales. Thus, traders' decision to buy or sell stocks might be influenced by the popularity of tweets' authors, not just tweets' content. We also did not group companies by sector. The literature findings show that this can improve predictive accuracy, particularly for IT, Finance, Media (Li et al. 2017) and Health (Shynkevich et al. 2016) industries.

Moreover, an important design choice that we made was grouping the tweets into independent event groups. The way we obtained aggregate group-level features from sentence-level features had a significant impact on the accuracy of our model. Due to time limitations, we only tested summing all the vectors' values in a group.

Feature extraction methods might have been impaired by presence of misspellings, which were not corrected. We attempted to eliminate misspellings with *tweetpreprocessor*, however we found this method to be too time-intensive to be practical for our dataset.

Lastly, we investigate only two views simultaneously with Canonical Correlation Analysis. CCA is based on the principle of maximising correlations in the common latent space. However, it is nontrivial to extend those methods to deal with multiple views, thus we did not attempt that given the time limitations.

7.4 Future Work

The identified limitations can be addressed with several extensions to the model design. Firstly, future research might take into account the popularity of authors of tweets and the popularity of the tweet itself. Several tweet meta-data tags are at a researcher's disposal, for example *retweet_count*, $favorite\_count$, *reply_count* to gauge the popularity of a tweet, as well as $followers\_count$ and $friends\_count$ to establish the popularity of the author. Filtering tweets based on some popularity threshold might reduce the noise in the dataset.

Secondly, future research might extend the dataset filtered only by the company username ("@Apple", not "Apple"). The analysis of the results have shown that this more rigorous method of tweet filtering leads to an increase in predictive accuracy from 64% to 67% for

seven-day return. Thus, future research might focus just on this method of data collection, perhaps expanding the timespan of the processed data to capture more tweets.

The text preprocessing could also be improved by correcting misspellings. Future researchers might use *tweetpreprocessor*'s built-in features or explore other off-the-shelf and custom-built methods.

Future research can group the companies in our dataset by industry. We have not done this due to time constraints.

Grouping of the tweets into independent events is a process that can undergo more scrutiny in the future projects. Perhaps averaging the vectors, or employing a more sophisticated grouping method would result in a better capturing of groups' tweets' semantics than the summation that we performed.

Lastly, multi-view learning approaches are broad, and some more sophisticated algorithms could be adopted to our problem in place of the CCA decomposition. For example, in order to take into account three views simultaneously, the future researches might employ Deep Semi-NMF as proposed by (Zhao et al. 2017).

Appendix A

Data Analysis Plots & Outputs

A.1 LDAvis graphs

In this section we present LDAvis graphs for all 10 themes found in the entire dataset.

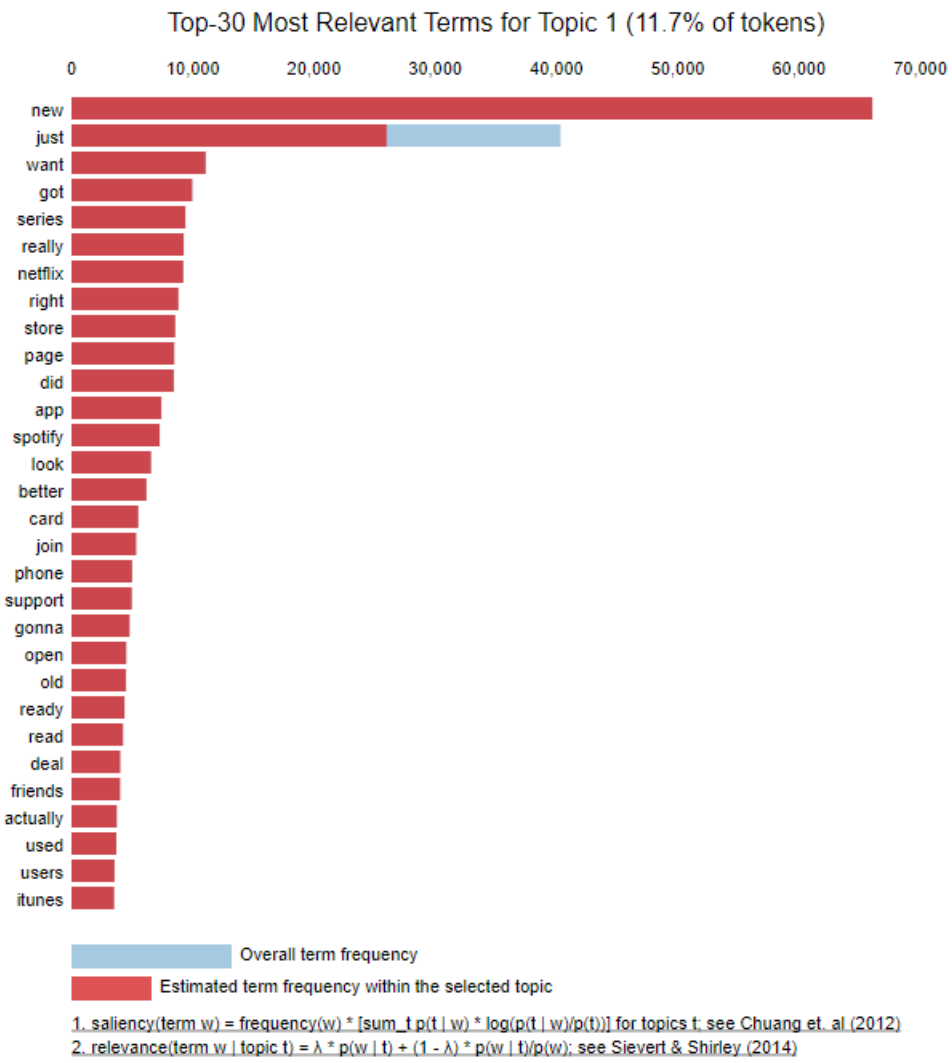

Figure A.1: Terms that are the most useful for interpreting Topic 1 from LDA

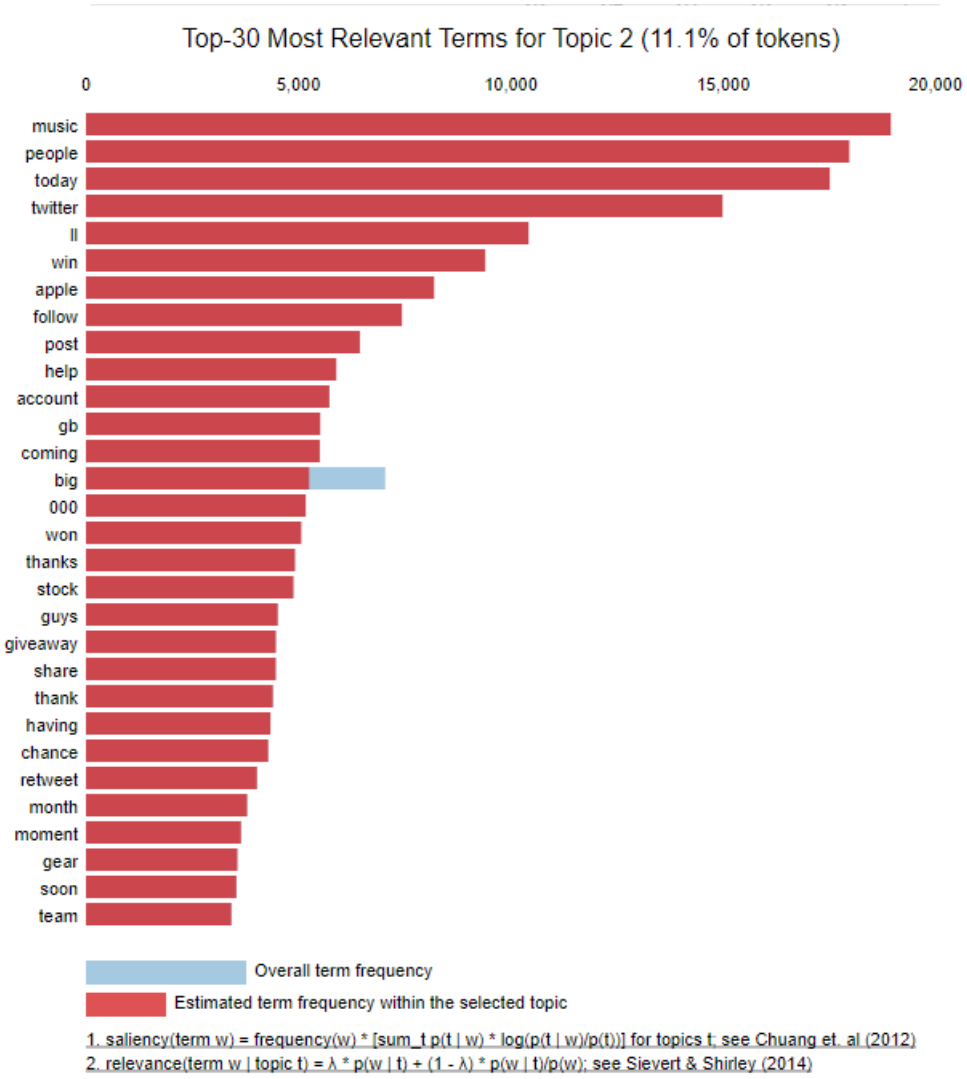

Figure A.2: Terms that are the most useful for interpreting Topic 2 from LDA

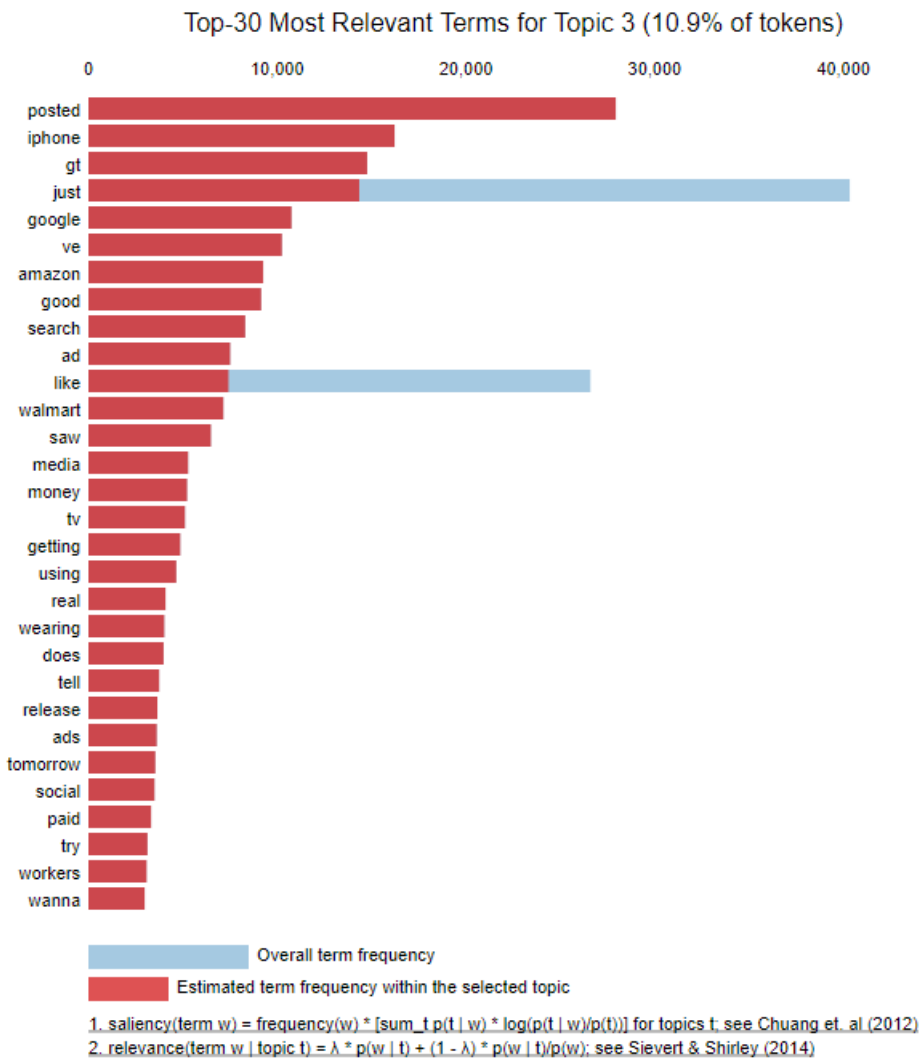

Figure A.3: Terms that are the most useful for interpreting Topic 3 from LDA

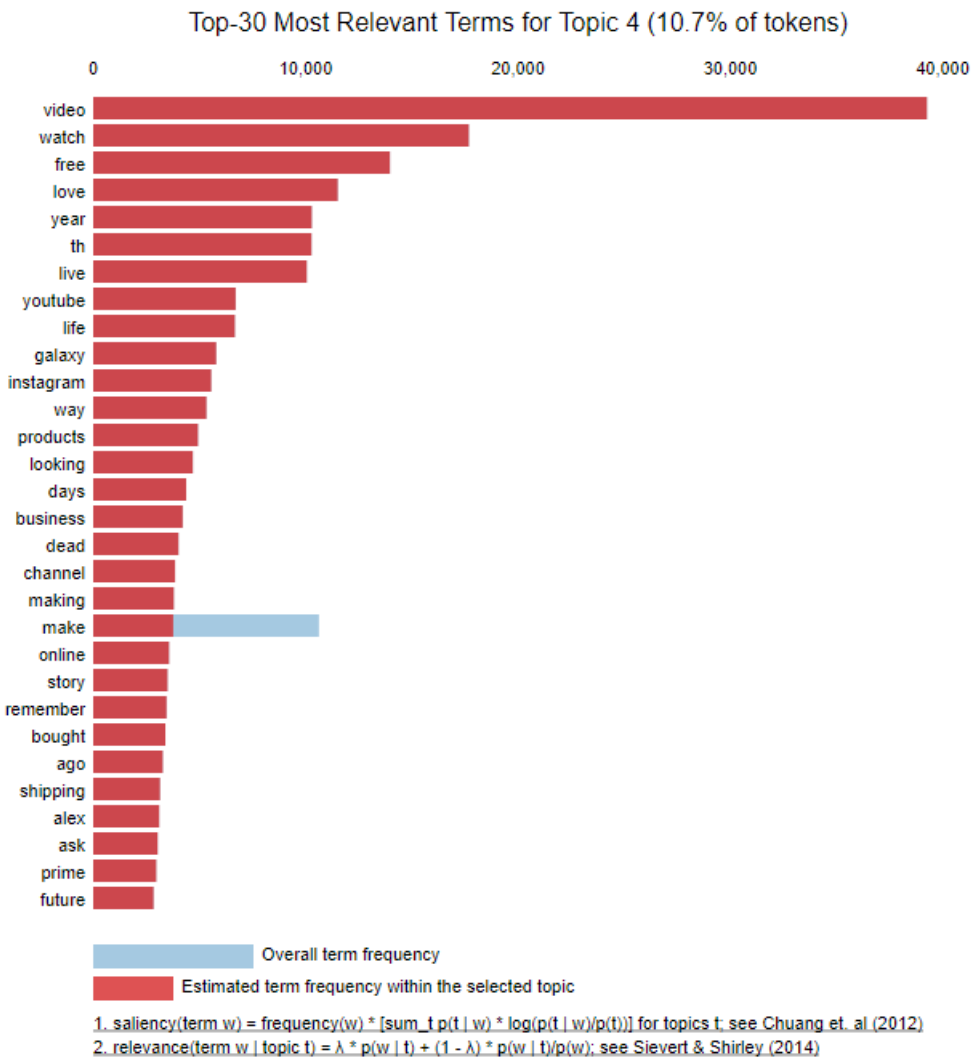

Figure A.4: Terms that are the most useful for interpreting Topic 4 from LDA

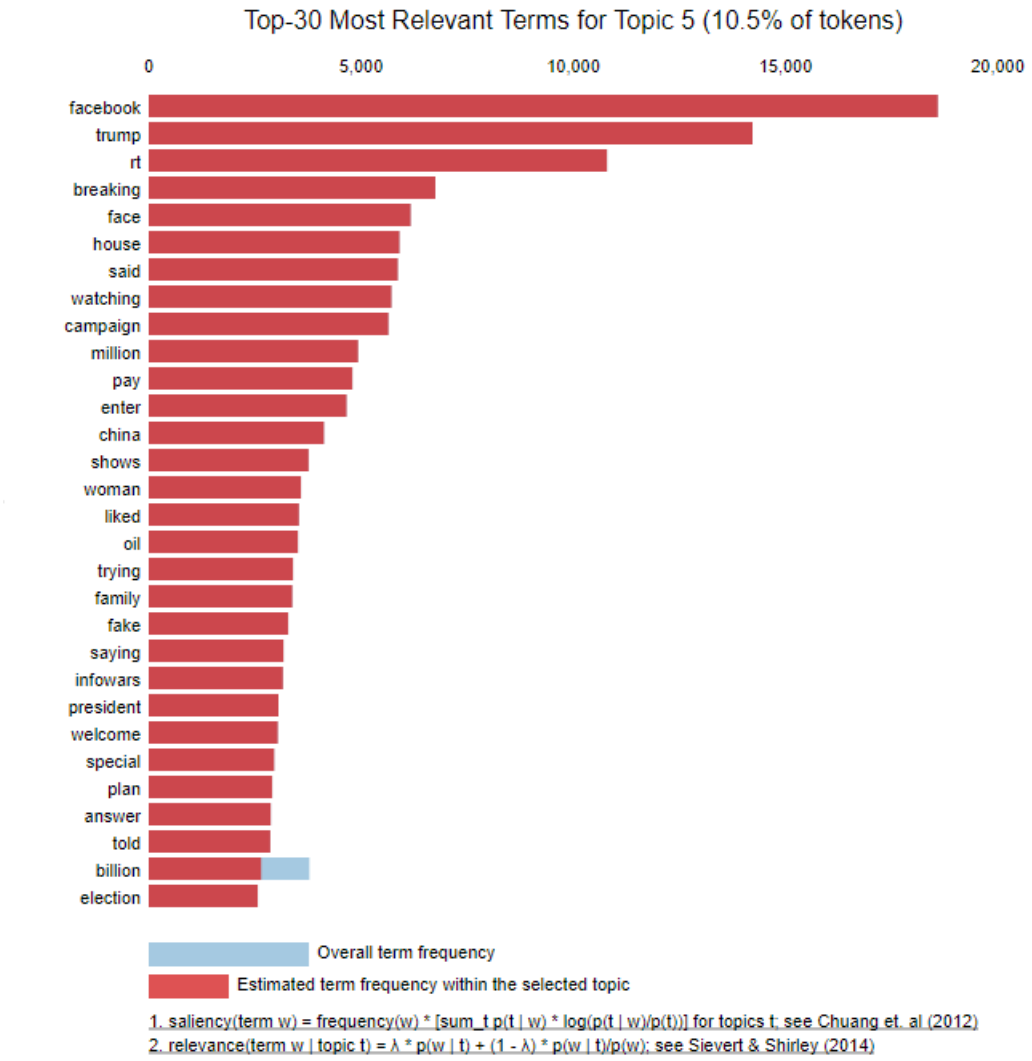

Figure A.5: Terms that are the most useful for interpreting Topic 5 from LDA

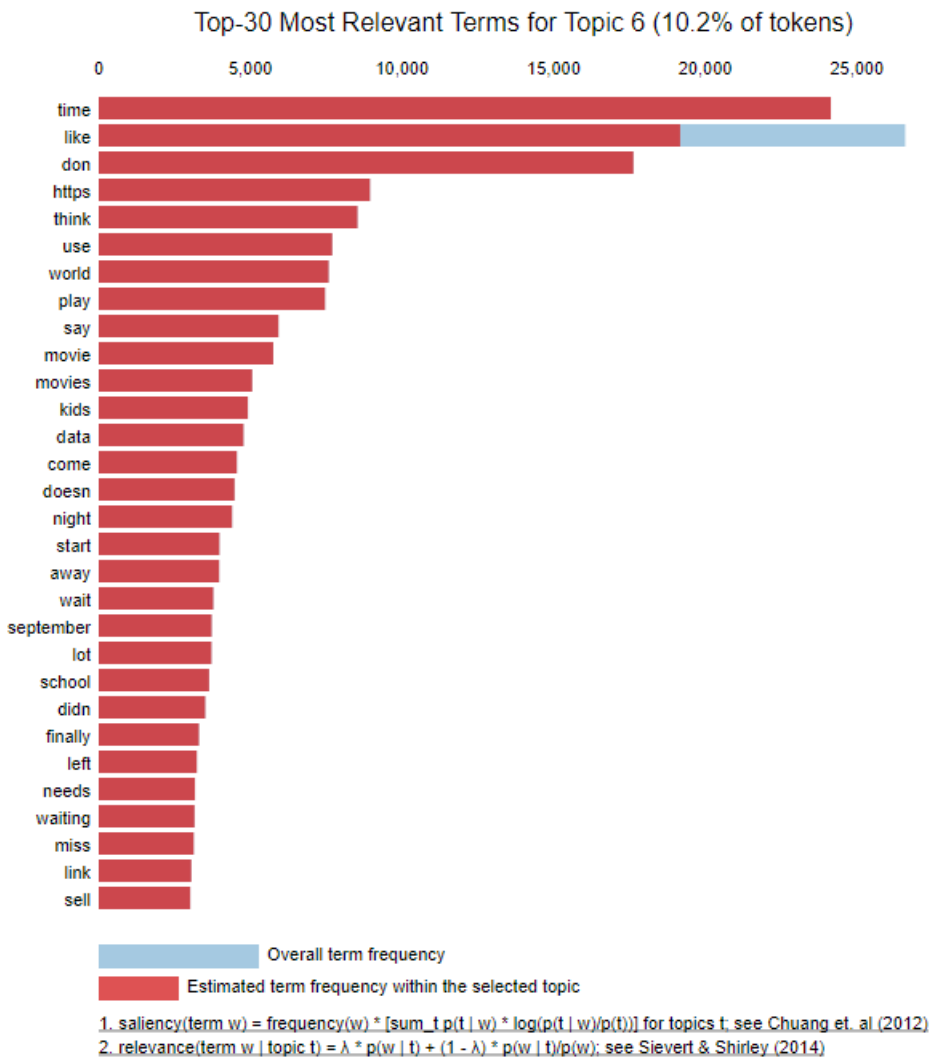

Figure A.6: Terms that are the most useful for interpreting Topic 6 from LDA

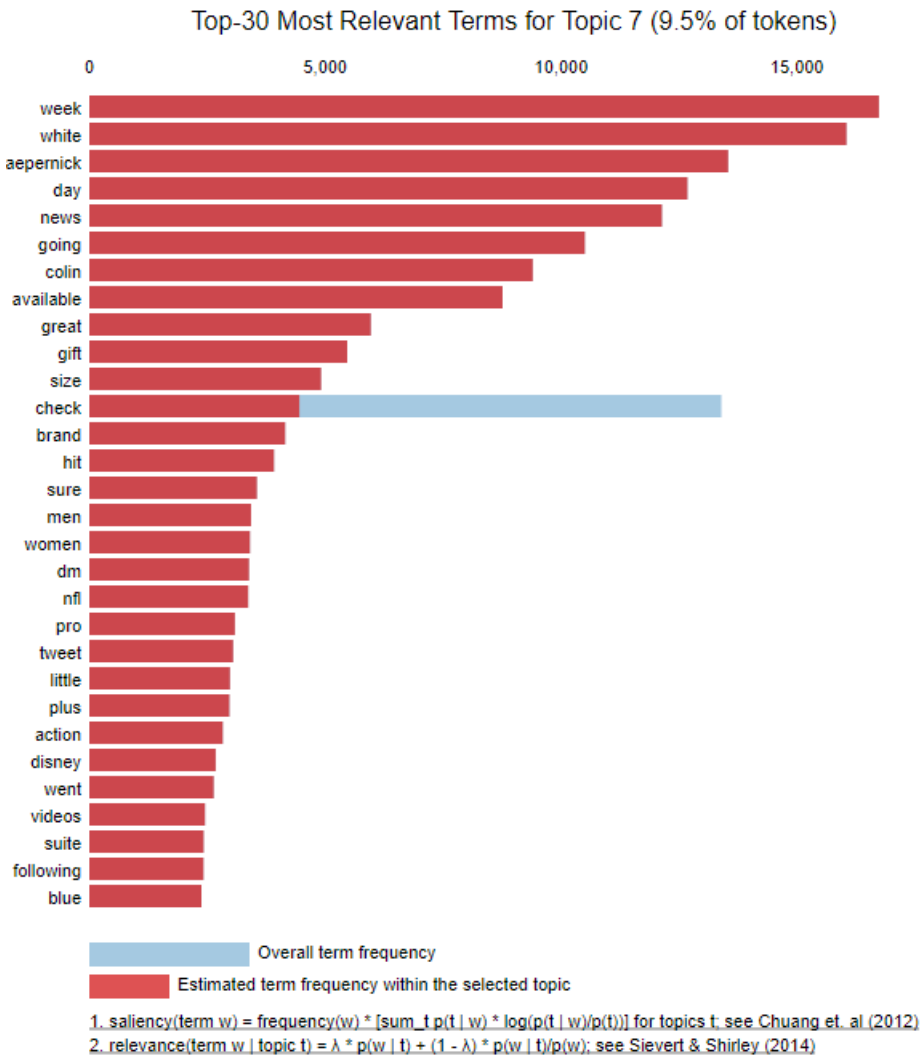

Figure A.7: Terms that are the most useful for interpreting Topic 7 from LDA

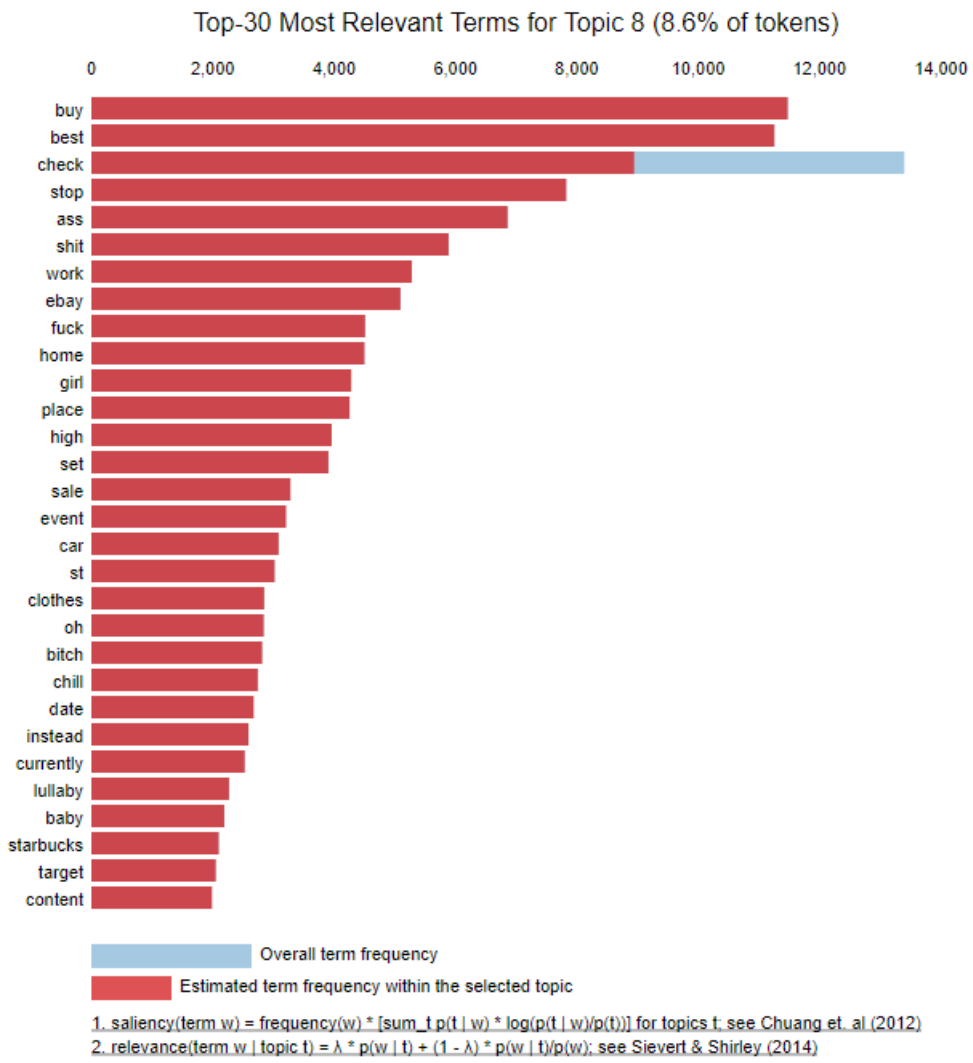

Figure A.8: Terms that are the most useful for interpreting Topic 8 from LDA

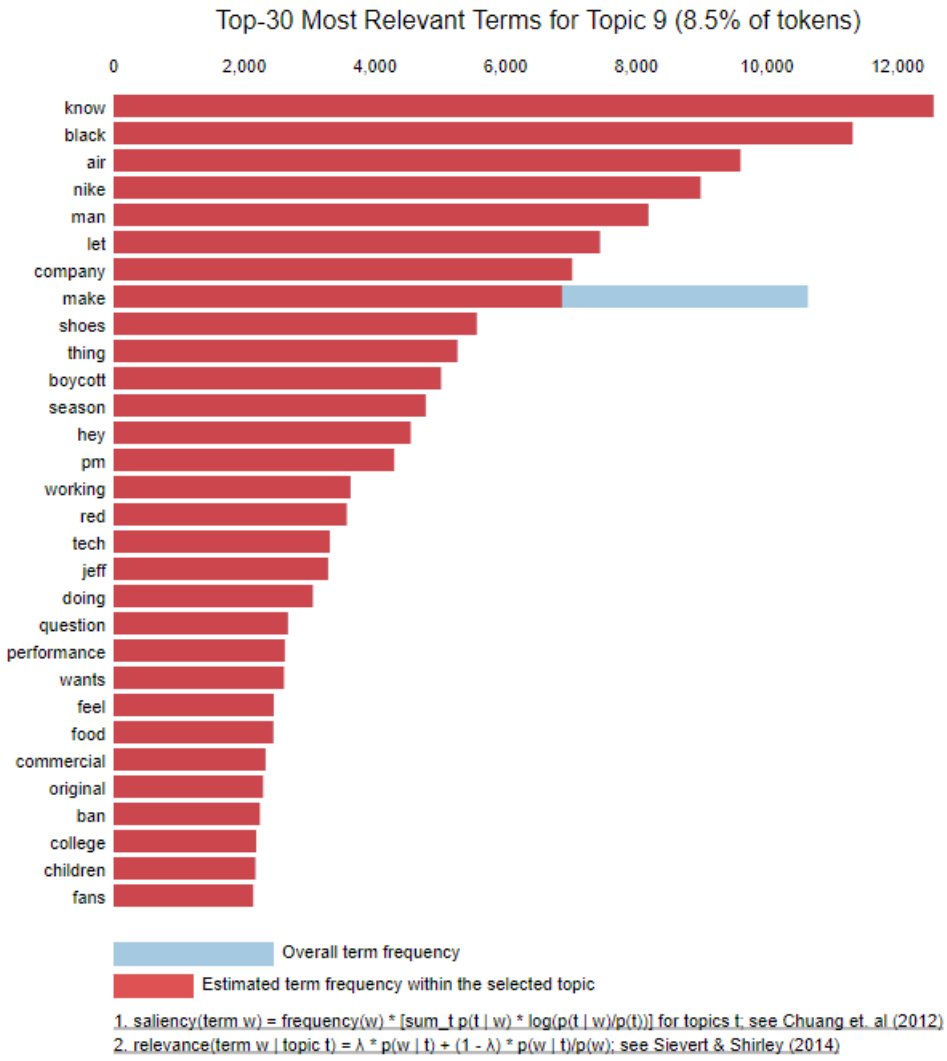

Figure A.9: Terms that are the most useful for interpreting Topic 9 from LDA

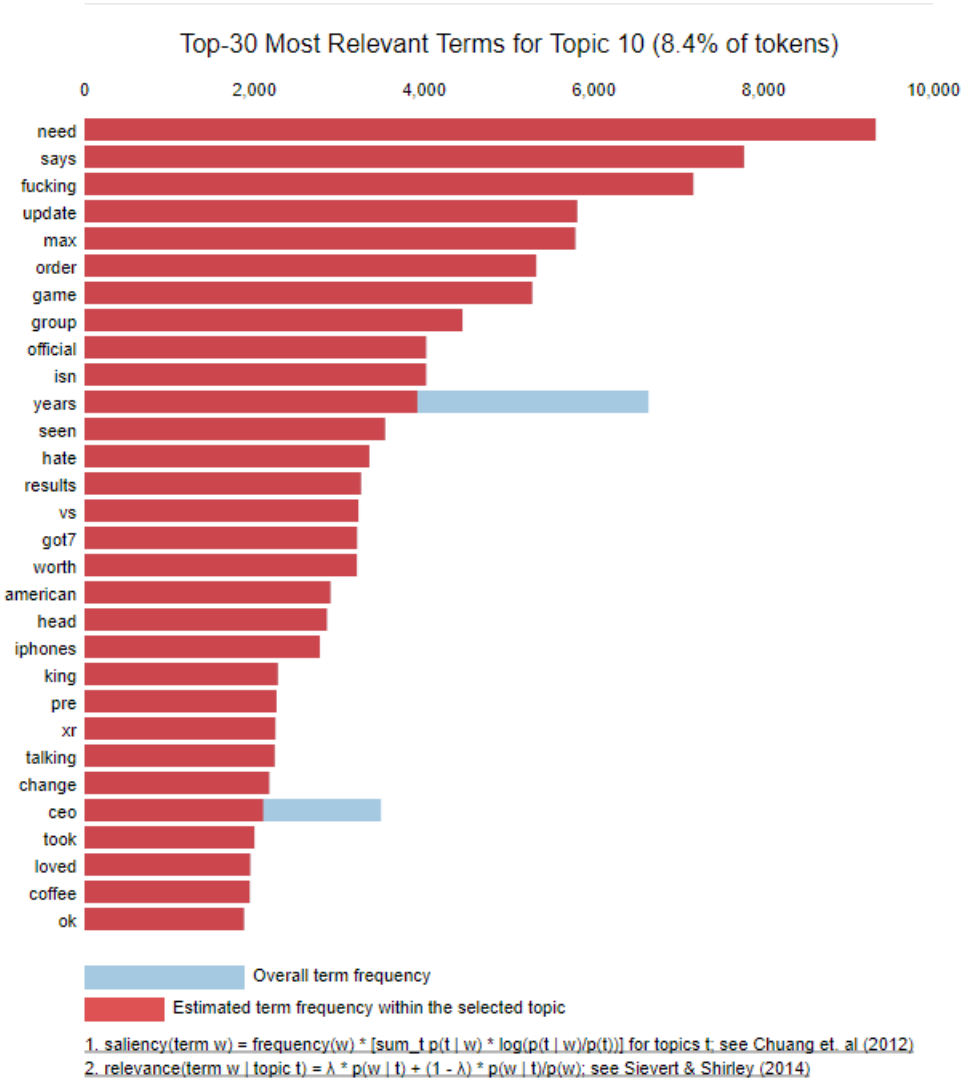

Figure A.10: Terms that are the most useful for interpreting Topic 10 from LDA

A.2 Miscellaneous

In Chapter 5 we mention that some tweets that we filter by keywords do not necessarily refer to the companies. The most spectacular example is the company Ford. After inspection of the most common words in the dataset, we find that majority of the tweets filtered by "Ford" keyword referred to the sexual allegations by Christine Blasey Ford against Brett Kavanaugh in 2018. This keyword constituted almost 11% of the entire dataset. We attach a sample of unprocessed tweets which revealed this noise:

	index	text	1DayReturn	2DayReturn	3DayReturn	7DayReturn
355550	779722	RT @AdamParkhomenko: Breaking: Ford will testi...	0.0	-0.004061	-0.007107	-0.040609
355551	779723	RT @BuckSexton: "Accepted" with conditions not...	0.0	-0.004061	-0.007107	-0.040609
355552	779726	RT @USANEWS007: BREAKING: Liberal quack, Profe...	0.0	-0.004061	-0.007107	-0.040609
355553	779728	RT @ChuckRossDC: More: Bromwich has resigned f...	0.0	-0.004061	-0.007107	-0.040609
355554	779729	BREAKING: Grassley Gives Christine Blasey Ford...	0.0	-0.004061	-0.007107	-0.040609
355555	779731	RT @EdKrassen: Some quotes survive 1000 yrs. \...	0.0	-0.004061	-0.007107	-0.040609
355556	779733	RT @KaivanShroff: MORE GOP HYPOCRISY:\r\r\r\...	0.0	-0.004061	-0.007107	-0.040609
355557	779734	RT @sxdoc: Woman who made false rape claim sen...	0.0	-0.004061	-0.007107	-0.040609
355558	779738	RT @SenKamalaHarris: Dr. Blasey Ford is willin...	0.0	-0.004061	-0.007107	-0.040609
355559	779740	RT @EdKrassen: Some quotes survive 1000 yrs. \...	0.0	-0.004061	-0.007107	-0.040609
355560	779741	RT @hrenee80: So, they've granted another exte...	0.0	-0.004061	-0.007107	-0.040609
355561	779742	RT @mikandynothem: Hollywood celebrities are v...	0.0	-0.004061	-0.007107	-0.040609
355562	779744	RT @chrislhayes: So even though Dr Ford is say...	0.0	-0.004061	-0.007107	-0.040609
355563	779746	RT @SenWarren: Hey, @SenMajLdr McConnell, @rea...	0.0	-0.004061	-0.007107	-0.040609
355564	779747	RT @kylegriffin1: New email from Dr. Ford's la...	0.0	-0.004061	-0.007107	-0.040609
355565	779750	RT @AriFleischer: 4) There should be no more ...	0.0	-0.004061	-0.007107	-0.040609
355566	779752	RT @TrumpsTaxes: @seungminkim They have comple...	0.0	-0.004061	-0.007107	-0.040609
355567	779754	RT @EdKrassen: Dear Dr. Christine Blasey Ford,...	0.0	-0.004061	-0.007107	-0.040609
355568	779759	RT @TheTestyTarheel: Well, looky here! \r\r\r\...	0.0	-0.004061	-0.007107	-0.040609
355569	779761	RT @ClaraJeffery: "Ford left the Washington ar...	0.0	-0.004061	-0.007107	-0.040609
355570	779763	RT @mitchellvii: So Christine Ford has agreed ...	0.0	-0.004061	-0.007107	-0.040609
355571	779764	RT @jrad1014hi: Ronald Reagan's Daughter Discl...	0.0	-0.004061	-0.007107	-0.040609

Figure A.11: An overview of tweets filtered by "Ford" keyword which do not relate to the automaker

A.3 Topic modelling

We present the full output from company-level topic modelling. We report results for 85 companies (15 companies in our dataset did not have any tweets mentioning them).

```
For company ASOS :
Topic 1: eh,follow,ends
Topic 2: rt,asos,voucher
Topic 3: work,time,nice

For company AT&T :
Topic 1: king,hey,tagged
Topic 2: donates,help,king
Topic 3: steve,go,get

For company Adobe :
Topic 1: photoshop,premiere,video
Topic 2: illustrator,new,premiere
Topic 3: react,graphic,photoshop

For company Allianz :
Topic 1: us,sao,arena
Topic 2: dying,brazil,stadium
Topic 3: see,season,arena

For company Amazon :
Topic 1: chance,gift,amazon
Topic 2: best,card,amazon
Topic 3: gift,via,amazon

For company American Express :
Topic 1: england,come,card
Topic 2: come,collab,new
Topic 3: make,spg,new

For company Apple :
Topic 1: watch,iphone,music
Topic 2: macbook,music,new
Topic 3: spotify,iphone,music

For company Audi :
Topic 1: tron,black,new
Topic 2: r8,new,car
Topic 3: tron,r8,audi

For company Aviva :
Topic 1: fund,us,community
Topic 2: billion,community,fund
Topic 3: community,us,vote

For company BMW :
Topic 1: guys,ran,rings
Topic 2: one,like,new
Topic 3: ran,bmw,choose

For company BP :
Topic 1: time,du,bp
Topic 2: no,du,bp
Topic 3: viewed,swift,ddu

For company Bank of America :
Topic 1: account,accounts,bank
Topic 2: america,account,bank
Topic 3: hey,style,united

For company Bayer :
Topic 1: toni,cancer,leverkusen
Topic 2: vidal,ingraham,throwback
Topic 3: deal,monsanto,leverkusen

For company BlackRock :
Topic 1: etf,eloring,coinbase
Topic 2: coinbase,etf,bitcoin
Topic 3: help,crypto,coinbase

For company Boeing :
Topic 1: air,two,named
Topic 2: boeing,big,air
Topic 3: spacex,new,flight

For company Burberry :
Topic 1: burberry,enter,lip
Topic 2: oxblood,burberry,fur
Topic 3: clothes,wearing,lip
```

```
For company CBS :
Topic 1: cbs,news,nbc
Topic 2: abc,news,cbs
Topic 3: nbc,moonves,news

For company Chevron :
Topic 1: much,dereg,trump
Topic 2: days,ross,trump
Topic 3: trade,public,tax

For company Cisco :
Topic 1: network,we,baby
Topic 2: we,ready,cisco
Topic 3: network,re,cisco

For company Citigroup :
Topic 1: aud,fully,inc
Topic 2: rating,week,crypto
Topic 3: week,crypto,new

For company CocaCola :
Topic 1: back,president,house
Topic 2: james,using,owner
Topic 3: juice,quincey,made

For company Colgate :
Topic 1: pipili,photo,hasthags
Topic 2: photo,palm,free
Topic 3: ako,ng,sachet

For company Comcast :
Topic 1: bid,sky,comcast
Topic 2: not,sky,fox
Topic 3: cable,comcast,fox

For company Costco :
Topic 1: ugh,costco,go
Topic 2: formula,day,samples
Topic 3: samples,costco,back

For company Danone :
Topic 1: smarties,tired,jig
Topic 2: tired,jig,mixed
Topic 3: r150,people,mixed

For company Disney :
Topic 1: love,not,movie
Topic 2: hey,one,favorite
Topic 3: disney,world,channel

For company Expedia :
Topic 1: away,star,night
Topic 2: night,365,eedia
Topic 3: a1,night,hotel

For company Exxon :
Topic 1: announced,regarding,exn
Topic 2: side,regarding,company
Topic 3: change,climate,exn

For company Facebook :
Topic 1: video,new,posted
Topic 2: page,facebook,video
Topic 3: facebook,video,new

For company FedEx :
Topic 1: ending,cup,using
Topic 2: player,week,cup
Topic 3: program,hiv,week

For company Gillette :
Topic 1: josh,tom,establish
Topic 2: gets,gordon,stadium
Topic 3: gordon,stadium,josh

For company Goldman Sachs :
Topic 1: shame,venezuela,bonds
```

```
Topic 2:
venezuela,dictatorship,regime
Topic 3:
venezuelan,financing,maduro

For company Google :
Topic 1: trump,google,mono
Topic 2: dropbox,search,google
Topic 3: rm,play,search

For company Groupon :
Topic 1: sm,stephen,groupon
Topic 2: great,ever,get
Topic 3: save,get,groupon

For company H&M :
Topic 1: fall,stay,new
Topic 2: scotland,tuned,28
Topic 3: part,excited,campaign

For company HP :
Topic 1: via,get,hp
Topic 2: gb,new,hp
Topic 3: hp,gb,laptop

For company HSBC :
Topic 1: bank,buhari,hsbc
Topic 2: hsbc,buhari,money
Topic 3: banking,hsbc,bank

For company Heineken :
Topic 1: watch,heineken,why
Topic 2: all,ad,drinking
Topic 3: lagos,fake,heineken

For company Home Depot :
Topic 1: driving,rented,manhattan
Topic 2: kills,terrorist,truck
Topic 3: kills,manhattan,driving

For company Honda :
Topic 1: new,accord,civic
Topic 2: car,like,civic
Topic 3: no,going,honda

For company Hyundai :
Topic 1: gay,girl,hyundai
Topic 2: sonata,driver,retweet
Topic 3: arab,new,sonata

For company IBM :
Topic 1: via,cloud,hat
Topic 2: new,red,blockchain
Topic 3: watson,cloud,ibm

For company Intel :
Topic 1: senate,committee,trump
Topic 2: trump,brennan,house
Topic 3: trump,committee,house

For company JPMorgan :
Topic 1: via,jpmorgan,chase
Topic 2: us,dimon,chase
Topic 3: ceo,says,chase

For company Kellogg's :
Topic 1: they,cdc,honey
Topic 2:
outbreak,cereal,salmonella
Topic 3: smacks,cdc,cereal

For company Kroger :
Topic 1: want,white,kroger
Topic 2: kroger,too,grocery
Topic 3: chef,commercial,boyardee

For company L'Oreal :
Topic 1: company,cruelty,free
Topic 2: law,means,says
```

```
Topic 3: required,cruelty,paris

For company Mastercard :
Topic 1: neymar,skill,day
Topic 2: day,outrageous,sends
Topic 3: sadio,picks,neymar

For company McDonald's :
Topic 1: homeless,fucking,like
Topic 2: burger,man,mcdonald
Topic 3: bitch,fries,mcdonald

For company Microsoft :
Topic 1: inserte,new,word
Topic 2: xbox,office,move
Topic 3: microsoft,windows,new

For company Morgan Stanley :
Topic 1: congratulat,deserv,done
Topic 2: congrats,morgan,well
Topic 3: chief,done,well

For company Nestle :
Topic 1: nestle,company,dollars
Topic 2: company,nestle,michigan
Topic 3: bottle,miles,water

For company Netflix :
Topic 1: rt,watching,netflix
Topic 2: season,netflix,watch
Topic 3: the,watching,put

For company Next :
Topic 1: next,stop,time
Topic 2: year,week,time
Topic 3: get,next,week

For company Nike :
Topic 1: shoes,air,kaepernick
Topic 2: air,colin,nike
Topic 3: people,colin,white

For company Nissan :
Topic 1: put,nobody,ain
Topic 2: altima,put,picking
Topic 3: picking,all,altima

For company Oracle :
Topic 1: concert,arena,oracle
Topic 2: via,fan,database
Topic 3: cloud,arena,oakland

For company P&G :
Topic 1: products,fox,toothpaste
Topic 2: advertises,start,know
Topic 3: news,lou,products

For company PayPal :
Topic 1: cash,not,follow
Topic 2: paypal,retweet,dm
Topic 3: get,cash,paypal

For company Pepsi :
Topic 1: kendall,drink,pepsi
Topic 2: it,pepsi,coke
Topic 3: like,coke,pepsi

For company Pfizer :
Topic 1: there,thank,making
Topic 2: making,year,reason
Topic 3: big,drugs,not

For company Reuters :
Topic 1: new,trump,journalists
Topic 2: years,reuters,trump
Topic 3: trump,myanmar,says

For company Ryanair :
```

```
Topic 1: racist,sit,elderly
Topic 2: racist,flight,black
Topic 3: woman,flight,man

For company SAP :
Topic 1: that,business,sap
Topic 2: technology,new,cloud
Topic 3: business,hana,sap

For company Samsung :
Topic 1: new,s8,galaxy
Topic 2: phone,note,galaxy
Topic 3: new,giveaway,note

For company Santander :
Topic 1: uncertainty,head,results
Topic 2: one,may,brexit
Topic 3: impact,significant,says

For company Shell :
Topic 1: oil,via,ghost
Topic 2: oil,today,shell
Topic 3: sea,shell,ghost

For company Siemens :
Topic 1: investment,pulling,comply
Topic 2: comply,us,iran
Topic 3: told,us,sanctions

For company Sony :
Topic 1: console,sony,playstation
Topic 2: ps4,not,new
Topic 3: games,new,playstation

For company Starbucks :
Topic 1: hot,make,dead
Topic 2: names,man,starbucks
Topic 3: all,fucking,coffee

For company TMobile :
Topic 1: world,donate,ll
Topic 2: home,every,hurricane
Topic 3: donate,we,tweet

For company Tesco :
Topic 1: buy,get,tesco
Topic 2: said,buy,please
Topic 3: said,shopping,people

For company Thales :
Topic 1: wants,report,spending
Topic 2: wants,spending,jobs
Topic 3: engineer,defense,west

For company Toyota :
Topic 1: going,details,we
Topic 2: share,red,going
Topic 3: details,red,share

For company TripAdvisor :
Topic 1: way,see,top
Topic 2: top,destination,need
Topic 3: top,reviews,trending

For company UPS :
Topic 1: bob,conditions,ups
Topic 2:
rescheduled,delivery,training
Topic 3: city,your,package

For company Verizon :
Topic 1: unlimited,service,verizon
Topic 2: phone,wireless,data
Topic 3: throttled,fire,data

For company Visa :
Topic 1: diversity,program,visa
Topic 2: gift,win,card
Topic 3: enter,gift,card
```

```
                                        Topic 2: talking,watching,customer
                                        Topic 3: stage,thinking,uses

For company Vodafone :
Topic 1: not,top,vodafone
Topic 2: rt,vodafone,first
Topic 3: still,network,song


For company Walmart :
Topic 1: white,dad,walmart
Topic 2: woman,like,walmart
Topic 3: stock,go,walmart


For company Wells Fargo :
Topic 1: tim,inappropriately,ll
Topic 2: using,math,need
Topic 3: inappropriately,tim,fire


For company Yahoo :
Topic 1: one,article,via
Topic 2: sports,via,yahoo
Topic 3: new,trump,news


For company adidas :
Topic 1: boost,available,free
Topic 2: black,video,lullaby
Topic 3: got7,available,yeezy


For company eBay :
Topic 1: via,new,check
Topic 2: gt,ebay,via
Topic 3: gt,via,check


For company easyJet :
Topic 1: still,delayed,flight
Topic 2: flight,delayed,no
Topic 3: cancelled,delay,flight


For company salesforce.com :
Topic 1:
software,salesforce,thinking
```

Bibliography

Abe, M. & Nakayama, H. (n.d.), Deep Learning for Forecasting Stock Returns in the Cross-Section, Technical report.

Agrawal, S., Jindal, M. & Pillai, G. (2010), Momentum analysis based stock market prediction using adaptive neuro-fuzzy inference system (anfis), *in* 'Proceedings of the international multiconference of engineers and computer scientists', Vol. 1.

Ahmed, W., Bath, P. A. & Demartini, G. (2017), Using twitter as a data source: An overview of ethical, legal, and methodological challenges, *in* 'The Ethics of Online Research', Emerald Publishing Limited, pp. 79–107.

Alammar, J. (2019), 'Visualizing machine learning one concept at a time - the illustrated transformer'.

Antoniou, C., Galariotis, E. C. & Read, D. (2012), 'Ambiguity Aversion, Company Size and the Pricing of Earnings Forecasts', *European Financial Management* **20**, 633–651.

Arora, S., Liang, Y. & Ma, T. (2016), 'A simple but tough-to-beat baseline for sentence embeddings'.

Ben-Ami, Z., Feldman2, R. & Rosenfeld, B. (n.d.), 'Using Multi-View Learning to Improve Detection of Investor Sentiments on Twitter'.

Bollen, J., Mao, H. & Zeng, X.-J. (n.d.), Twitter mood predicts the stock market, Technical report.

Campbell, J. Y., Lo, A. W.-C., MacKinlay, A. C. et al. (1997), *The econometrics of financial markets*, Vol. 2, princeton University press Princeton, NJ.

Cooper, M. J., Jackson Iii, W. E. & Patterson, G. A. (2003), 'Evidence of predictability in the cross-section of bank stock returns', *Journal of Banking & Finance* **27**, 817–850.

De Bondt, W. F. & Thaler, R. (1985), 'Does the stock market overreact?', *The Journal of finance* **40**(3), 793–805.

Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. (2018), 'Bert: Pre-training of deep bidirectional transformers for language understanding', *arXiv preprint arXiv:1810.04805* .

Fama, E. (1970), 'Efficient capital markets: A review of theory and empirical work', *The Journal of Finance, 25(2), p. 383* .

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M. & Hutter, F. (2015*a*), Efficient and robust automated machine learning, *in* C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama & R. Garnett, eds, 'Advances in Neural Information Processing Systems 28', Curran Associates, Inc., pp. 2962–2970.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M. & Hutter, F. (2015*b*), Efficient and robust automated machine learning, *in* 'Advances in neural information processing systems', pp. 2962–2970.

Girolami, M., Kabán, A. & Kabán, K. (n.d.), On an Equivalence between PLSI and LDA, Technical report.

Li, B., Chan, K. C., Ou, C. & Ruifeng, S. (2017), 'Discovering public sentiment in social media for predicting stock movement of publicly listed companies', *Information Systems* **69**, 81–92.

Li, X., Huang, X., Deng, X. & Zhu, S. (2014), 'Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information', *Neurocomputing* **142**, 228–238.

Mehrara, M., Moeini, A., Ahrari, M. & Ghafari, A. (2010), 'Using technical analysis with neural network for forecasting stock price index in tehran stock exchange', *Middle Eastern Finance and Economics* **6**(6), 50–61.

Oliveira, N., Cortez, P. & Areal, N. (2017), 'The impact of microblogging data for stock market prediction: Using Twitter to predict returns, volatility, trading volume and survey sentiment indices', *Expert Systems with Applications* **73**, 125–144.

Patankar, P. M. & Swati (2014), 'Short term stock selection with case-based reasoning technique', *Applied Soft Computing* **22**, 205–212.

Phua, P. K. H., Zhu, X. & Koh, C. H. (2003), Forecasting stock index increments using neural networks with trust region methods, *in* 'Proceedings of the International Joint Conference on Neural Networks, 2003.', Vol. 1, IEEE, pp. 260–265.

Roondiwala, M., Patel, H. & Varma, S. (2015), Predicting Stock Prices Using LSTM, Technical report.

Shynkevich, Y., McGinnity, T., Coleman, S. A. & Belatreche, A. (2016), 'Forecasting movements of health-care stock prices based on different categories of news articles using multiple kernel learning', *Decision Support Systems* **85**, 74–83.

Sievert, C. & Shirley, K. (2014), Ldavis: A method for visualizing and interpreting topics, *in* 'Proceedings of the workshop on interactive language learning, visualization, and interfaces', pp. 63–70.

Trejo Pech, C. O., Noguera, M. & White, S. (2015), 'Financial ratios used by equity analysts in Mexico and stock returns', *Contaduría y Administración* **60**(3), 578–592.

Xu, C. & Xu, C. (2013), A Survey on Multi-view Learning, Technical report.

Yu, H., Chen, R. & Zhang, G. (2014), 'A SVM Stock Selection Model within PCA', *Procedia Computer Science* **31**, 406–412.

Zhang, X., Zhang, Y., Wang, S., Yao, Y., Fang, B. & Yu, P. S. (2018), 'Improving stock market prediction via heterogeneous information fusion', *Knowledge-Based Systems* **143**, 236–247.

Zhao, H., Ding, Z. & Fu, Y. (2017), Multi-view clustering via deep matrix factorization, *in* 'Thirty-First AAAI Conference on Artificial Intelligence'.

Zhou, X., Pan, Z., Hu, G., Tang, S. & Zhao, C. (2018), 'Stock Market Prediction on High-Frequency Data Using Generative Adversarial Nets'.