Department of Computer Science
Faculty of Engineering, University of Freiburg

Bachelor's Thesis

Who drives the market? Sentiment analysis of financial news posted on Reddit and Financial Times

By
Michael Lubitz

Date of Submission
07.11.2017

Reviewer
Prof. Dr. Hannah Bast

Supervisor
Prof. Dr. Dirk Neumann

Declaration

I hereby declare, that I am the sole author and composer of my Thesis and that no other sources or learning aids, other than those listed, have been used. Furthermore, I declare that I have acknowledged the work of others by providing detailed references of said work. I hereby also declare, that my Thesis has not been prepared for another examination or assignment, either wholly or excerpts thereof.

_____ _____

Place, Date Signature

Abstract

In this thesis, we evaluate whether the sentiment in financial news articles posted on Reddit can predict future stock market returns. For this purpose, we extracted all submissions that have been posted on the *economics* subreddit between January 2008 and July 2017. We evaluate the predictive power of Reddit by comparing the sentiment of all submitted articles posted on a trading day to the corresponding closing value of the S&P 500 stock index. The sentiment in these articles is calculated using a dictionary-based approach and supervised machine learning models based on different feature extraction methods. In addition, we perform a weighted sentiment calculation by incorporating the number of votes and comments for Reddit submissions into our models. News published in the Financial Times will act as a baseline. Our results suggest that the predictive power of Reddit is slightly better than using standard news paper analysis. We achieve 56.49% of accuracy in predicting the future direction of the stock market which corresponds to the results of recent academic research.

Zusammenfassung

In dieser Arbeit gehen wir der Frage nach, ob es möglich ist anhand der Stimmung von Finanznachrichten, welche auf Reddit geteilt wurden, auf Veränderungen eines Aktienindexes zu schließen. Dazu untersuchen wir Nachrichten, welche zwischen Januar 2008 und Juli 2017 im Subreddit *economics* veröffentlicht wurden. Anschließend betrachten wir die Stimmung aller Newsartikel und erstellen anhand dessen für jeden Handelstag ein Stimmungsbild. Daraus leiten wir eine Vorhersage ab, wobei wir bei einer positiven Stimmung eine Aufwärtsbewegung des Aktienmarktes vermuten und umgekehrt bei einer negativen Stimmung. Daraufhin vergleichen wir den vorhergesagten Trend mit dem wahren Wert am jeweiligen Handelstag. Als Aktienindex dient uns hierbei der S&P 500 Index aus den USA. Um die Stimmung abzulesen, verwenden wir zwei Methoden. Die erste Methode basiert auf einem Wörterbuch, welches sowohl positive als auch negative Wörter beinhaltet. Die zweite Methode verwendet verschiedene Machine Learning Algorithmen. Darüberhinaus testen wir Möglichkeiten, Stimmungswerte anhand der Anzahl von Kommentaren und Votes zu gewichten. Unsere Ergebnisse vergleichen wir anschließend mit einer Analyse von Financial Times Nachrichten aus dem selben Zeitraum. Dadurch lassen unsere Ergebnisse, mit bis zu 56.49% Genauigkeit, vermuten, dass Reddit eine etwas höhere Vorhersagekraft im Vergleich zu einer Standard Nachrichtenanalyse besitzt.

Contents

1 Introduction

Predicting the movement of stock market prices has always had a certain appeal to academic research. According to the *efficient market hypothesis* (EMH) [1], it is impossible to use publicly available information to yield higher profits, as these information become part of the market in the very moment they are shared. Following this hypothesis, many experts argue that stock markets follow a random walk pattern [2, 3], which makes it an extremely difficult task to predict price movements precisely. However, some research has successfully challenged the efficient market hypothesis by investigating stock markets in some countries, e.g. [4, 5]. As already stated by Thomas and Sycara [6], our concern is not the exact prediction of future values but rather a prediction whether a stock index moves up or down, since traders and economists are more interested in the profitability of trading rules.

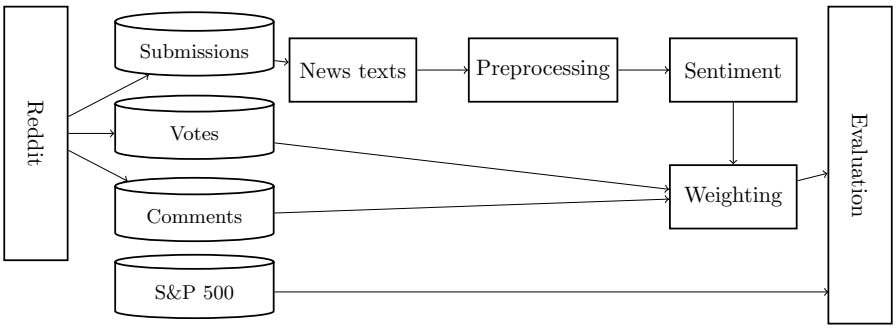

Figure 1: Framework to evaluate predictions based on textual sentiment analysis

In this thesis we introduce and evaluate different methods to predict stock market movements using financial news posted on the online bulletin board Reddit. Specifically, we analyze all submissions in the subreddit *economics* that have been posted between January 2008 and July 2017. The setup of our framework to calculate and evaluate predictions is displayed in Figure 1. We determine the sentiment in these news articles and investigate the predictive power of them. To do so, we use a dictionary containing business related words, which are labeled either as positive or negative, as well as different machine learning techniques, namely Naive Bayes, Random Forests and Support Vector Machines. For each of these techniques we evaluate different methods to extract numerical feature vectors from the textual data. In a next step, we investigate the meaning of votes and comments in terms of the relevance of submitted news texts. Based on these prospects, we derive weighting methods for the sentiments of texts.

After that, we evaluate our results by comparing them to a baseline, which consists of Financial Times articles. To retrieve all the necessary data for this thesis, we wrote several web crawling programs using Python. After analyzing the sentiments in the collected news articles our target variable is given by the direction of the stock market on the corresponding trading day (i.e. positive or negative). Our prediction is correct if the stock index closes with a positive (negative) change and our calculated sentiment value on this day is positive (negative) as well. If the index closes steadily, we consider this as a positive closing since this would not cause any losses.

Following these descriptions, we achieve an accuracy of 56.49% in predicting the future direction of the stock market. Furthermore, we find our predictive accuracy to be higher than by analyzing financial news alone. In addition, our results suggest that the voting and comment system of Reddit has the change to increase the predictive performance.

The remainder of this thesis is structured as follows. Section 2 gives an overview of related work and enumerates recent results in academic research. In Section 3 describes our data sources and how the data was obtained. Section 4 details the methods we use for the sentiment calculation. Our results are presented in Section 5. Section 6 discusses our findings and Section 7 concludes this thesis.

2 Related Work

In this section, we will discuss some recent approaches in the field of stock market prediction based on textual sentiment analysis. Due to the appearance of many new data sources the research has changed tremendously over the last years. There are two common methods for predicting future stock market returns, namely technical and fundamental analysis.

2.1 Traditional Sources

As stated by Blume, Easley and O'hara [7] technical analysts believe that future price movements can be predicted by analyzing price and volume data. Pring [8] details the use and the role of technical analysis in his work. However, the accuracy of this technique is discussed controversially by economists and is considered to be less successful by Gidófalvi [2].

On the other hand there is fundamental analysis, the approach which we follow in this thesis. Fundamental analysts base their predictions on factors in the real economy (e.g. inflation, trading volume, organizational changes in a company), as described by Gidófalvi [2]. He states financial online news as a source of indicators which potentially contain useful information for fundamental analysts. Furthermore, financial news articles are considered as the major source of market information for investors [9]. Cecchini et. al. [10] found that financial news can

have the power to predict financial events (i.e. bankruptcy and fraud). Thus, many researchers have tried to figure out if one can transfer this predictive power to the stock markets. Niederhoffer [11] analyzed the effect of *New York Times* headlines on the Standard and Poor's Composite Index. He pointed out that markets tend to overreact to negative headlines on the first and second day after a world event took place. Wüthrich et. al. [12] managed to achieve an accuracy of 45% on predicting the direction of the Dow Jones Industrial Average using online financial news. They created a system that uses a three class prediction (up, down or steady) based on old financial news texts combined with previous stock values as a training set. Schumaker and Chen [13] achieved a predictive power of 58% on values of the S&P 500 stock index using a similar training set as described by Wüthrich et. al. to train Support Vector Machines. They found that extracting nouns from news texts leads to a more accurate result compared to a bag-of-words representation. Ammann, Frey and Verhofen [14] examined the *Handelsblatt*, a leading German financial newspaper, and stated that newspaper content in general do have the power to predict future price movements. They used the DAX as a target for their predictions. Although, all these researchers stated that news most certainly have influence to the stock market, we must not underrate the effect of public mood stages and sentiment.

Psychological researchers like Dolan [15] state that emotion biases decision making, particularly in making decisions under risk [16]. Naturally, this also affects investors, traders and other financial decision-makers as pointed out by Nofsinger [17]. From his investigations, Nofsinger concluded that the stock market itself is a measure of social mood, whereupon a high (low) stock market level indicates a high (low) social mood. Since the appearance of social media platforms like Twitter it has never been easier to analyze public mood. Nowadays, almost everybody can share his opinion online and researchers are then able to use this data.

2.2 Social Media

Since we are using Reddit to retrieve our collection of financial news articles, we also have to understand the influence of the mood in social networks on stock price changes. A large amount of research has been done in this field recently. Bollen and Mao [18] were able to bring the mood of people posting on Twitter in relation with the behaviour of the stock market. They used different tools to classify about ten million collected Twitter messages by approximately 2.7 million users into six stages of mood, namely calm, alert, kind, vital, sure and happy. Then they used the classified Twitter messages combined with the corresponding stock market values (Dow Jones Industrial Average) as training set for a neural network. Their model achieved an accuracy of 86.7%. Mittal and Goel [3] approved the results of Bollen and Mao in their own work. Furthermore, they were able to show a correlation between calm and happy mood stages with Dow Jones Industrial Average values. It can be observed that the use of Twitter mood leads to a higher accuracy in predicting stock market behaviour than by analyzing financial news. We will now evaluate, whether a combination of both fields is reasonable to predict stock prices.

3 Data Sources

This chapter describes the data sources we are using in our analysis. The first section briefly describes the platform Reddit and how we extracted the submission data. The following section does the same for the Financial Times, which we later use to compare the accuracy of Reddit. The third section describes the S&P 500 data and the last section the 8K data which we use as a training set for our machine learning algorithms.

3.1 Reddit.com

Reddit.com, founded in 2005, is a social news aggregator with 274 million unique monthly visitors in January 2017 according to Reddit [19]. The platform entitles itself *the front page of the internet*. Each registered user can share links to online news (or other contents) with the community. This is called a *submission*. Other users are then able to vote (either up or down) on the relevance and importance of a submission [20]. Besides voting, users can also comment on a posted topic and discuss its content.

Reddit as a large community is divided into sub-communities, so called *subreddits*. Each subreddit is addressed to its own topic. There are subreddits for political discussions, for music enthusiasts, for football fans, and many more. Each registered user can create his own subreddit. These subreddits are maintained and

supervised by moderators, who are installed by the creator of a subreddit. They also specify rules for which topics are allowed to be posted and which are not.

Each subreddit contains five sections, namely hot, new, rising, controversial, and top. These sections contain pages, which again contain 25 posts by default. The *new* section displays all posts according to the time they have been created in a reverse chronological order. So every time a user creates a new submission it will appear on the top of the *new* section, but will also push the last submission from the first page of the section. The *rising* section features all posts that are currently getting a lot of positive votes, whereas the *controversial* section displays posts that have gathered a similar amount of positive and negative votes. The *top* section contains posts that have the most absolute number of positive votes in a set time period. The last section, *hot*, is the default view a user gets if he enters a subreddit. It displays all submissions that are getting a lot of votes and comments recently.

Whoever wants to use Reddit as a source of data for research has to understand the dynamic of submissions and votes. The number of votes determines the position, and therefore the visibility of a submission on the page (apart from *new*), assuming that users will most certainly only visit the first pages of a section. Since all submissions start in the *new* section they all have the same chance of getting votes and gaining popularity. This fact also implicates that a submission loses the focus of the community while it keeps getting pushed down further by subsequent submissions. In turn, submissions with a large amount of up-votes will gain more visibility, since they get featured in the *hot* section, which again leads to more votes, following the phrase "rich get richer". This also means that it is possible that eventually valuable posts get overlooked and buried by the community. Gilbert [21] found out that 52% of the most popular links (after they have been posted again) have been overlooked at the first time they were submitted. Horne and Adali [20] pointed out that voting is based on content quality, whereas commenting

is based on emotion. Following these findings with reference to the dynamics of Reddit, we investigate if a weighted sentiment calculation with respect to comments and votes is reasonable.

Based on its characteristics, we consider Reddit not only as a content distributor, but also as a content filter, since its users determine which content ends up on Reddit and which content stays on Reddit. Buntain and Golbeck [22] observed that most of the Reddit users contribute to one single subreddit only. Hence, we assume that most of the active contributors to a specific subreddit are at least familiar with its respective topic. In addition, users determine which news they consider to be relevant by voting on these submissions. This clearly differs from classical news websites, where editors choose which content appears where (e.g. on the front page as the main article) and which content appears at all.

As we need business and economy related news, we chose the subreddit *economics* as our main source of financial news. We evaluated many different subreddits with similar topics, but find that *economics* fits best for our intention. The administrators ensure that users submit links to economic valuable sources only. They also demand from their moderators to have a solid background in economics. Thus, we assume that this subreddit is a reliable source for our data.

To retrieve the required data, we have written different programs using Python. It turned out that it is a quite difficult task to obtain submissions that are older than a few weeks, since they disappear from the website after some time. Even though Reddit maintains a Application Programming Interface (API), we have to deliver the ID of a submission to be able to retrieve data via the API. We tried to use the *Wayback Machine*[1], but observed that this archive is rather fragmentary, which makes it useless for our purpose. We then found a way to manipulate the search form on the *ecomomics* subreddit, to use it as an archive for

[1]http://web.archive.org, last lookup August 03, 2017

its submissions. We have written a web crawler that iterates over all submissions and pages, and stores the submission IDs in a database. After that we used the Python package `PRAW`[2], which in turn uses the Reddit API, to retrieve all submission data. This data includes the submission title, a link to an online resource, the date the submission was published on, the absolute score of votes and the number of comments. As a next step, we have written a second crawler which is able to download the news texts from the linked websites. To achieve this, we had to write rules for each website which state in which HTML element the text is located. Due to the large amount of different websites (12,086) we had to restrict ourselves to the 100 (which are 46% of all posted news) most common websites in our link collection. This reduces the quantity of news texts, but it would be impossible to write rules for every single website to extract only the actual article content. Furthermore, this gives us the facility to remove noise from the texts (e.g. advertisements or titles of related news). We end up with 43,205 articles from January 2008 to July 2017 and obtained several different news for each trading day.

3.2 The Financial Times

To evaluate the significance of our results, we apply the same methods we used to determine the sentiment of the news we obtained through Reddit on news texts by the Financial Times. We chose them as a baseline, since the Financial Times is considered as an important source for independent business related news [12].

To obtain the news from the Financial Times website[3] we have written another crawler, that iterates over the news archive of the Financial Times and downloads all news data, including the news text and the date of publication. Using this method, we were able to collect 592,689 Financial Times articles published

[2]https://github.com/praw-dev/praw, last lookup August 24, 2017
[3]https://ft.com, last lookup on August 13, 2017

between January 2008 to July 2017.

3.3 Stock Price Data

We use Standard and Poor's 500 (S&P 500) values from January 2008 to July 2017, obtained from Yahoo! Finance. The data includes the opening and closing values for each day in this time period. We calculated the change for every trading day as the difference between closing and opening value. The S&P 500 stock market index, which is grounded on the market capitalization of 500 companies listed on American stock exchanges, is considered as one of the most important financial indicators worldwide [23]. We chose this index as the U.S. is a remarkable financial center and is one of the largest economies in the world. Furthermore, 54% of Reddit's visitors come from the United States according to Reddit [19].

Our stock data contains the financial crisis beginning in 2008. Hence, we have many declines in prices, which make our predictions more difficult, but also more valuable. The stock data is distributed as shown in Table I.

Table I: Distribution of S&P 500 directions after closing

Positive	Negative	Total
1096	1297	2393

3.4 8K News (Training Set)

We use this collection of financial news texts as a training set for the machine learning algorithms we are using to determine sentiment values. It has been handed out by the Information Systems Research chair of the University of Freiburg and contains already labeled news texts. Thus, we did not have to label texts for

the training set on our own. Labeling texts is an extensive and time-consuming task and our labels would be biased by our personal attitude and would be highly subjective. Each text in the 8K set is labeled with the *abnormal return* value of the company the news is subjected to.

The abnormal return AR_{ct} is the difference between the actual return R_{ct} and the expected return $\mathrm{E}(R_{ct})$ for a company c on day t as described in the following equation:

$$AR_{ct} = R_{ct} - \mathrm{E}(R_{ct}) \tag{1}$$

In stock market trading the expected return of a company in a stock index is the increase or decrease value of the whole index. The actual return is the increase or decrease value of the company's stock. Therefore, we consider a text labeled with a positive (negative) abnormal return value to be positive (negative)

4 Sentiment Analysis - Methodology

This section describes the methods we use to analyze the sentiment of our news text collection. First, we evaluate the use of a dictionary-based method to determine the sentiment in texts. Second, we detail the use of traditional machine learning models trained on the 8K dataset.

4.1 Sentiment Analysis Using Dictionaries

Our first approach uses a dictionary of words. This dictionary, containing positive and negative words, was created by Loughran and McDonald [24] in order to analyze the tone of financial news texts. An excerpt of this dictionary is shown in Table II.

Table II: Excerpt of the sentiment dictionary

Word	Sentiment
accomplish	positive
effective	positive
perfect	positive
bankruptcy	negative
failure	negative

Before we are able to analyze the sentiment of our collected news, we need to preprocess the unstructured textual data as follows:

1. Remove all special characters and line breaks, and then convert the whole text to lowercase.

2. Split each text into a list of words and remove all common stop words (e.g. the, a, for) from this list.

3. Remove all words containing less than three characters from the list for performance purposes in the calculation step. Removing these words will not affect the analysis since we do not have words with less than three characters in the sentiment dictionary.

Following the concept of text classification, we will consider a news text as a *document* d from now on. Since our goal for this particular step is to decide whether a document is either considered to be positive or negative, our model assigns a sentiment value s_d in the range $[-1, 1]$ to each document d in our collection. A positive (negative) number means a positive (negative) tone. The sentiment of a document is calculated as described in equation (2). We calculate the difference between the number of positive p and negative n words (or matches) and divide it by the total number of positive and negative words contained in the document. This means our declaration is based on which class of words (positive or negative) is located more frequently in a document. A sentiment value of zero indicates that either we were not able to find a match in a text, or that the amount of positive equals the amount of negative matches. The tone of a document with a sentiment value of zero could be interpreted as *neutral*.

$$s_d = \begin{cases} \frac{p-n}{p+n}, & \text{if } p+n > 0 \\ 0, & \text{else} \end{cases} \tag{2}$$

4.2 Sentiment Analysis Using Machine Learning

Our second approach to analyze the sentiments of the collected news texts is by using *supervised machine learning* techniques. Pang et. al. [25] stated that it seems relatively easy for humans to distinguish positive from negative news. For machines this is a rather difficult task since humans connect strong sentiments with some specific words. However, sentiment is strongly subjective, and, therefore, it is necessary to use a solid, neutral and traceable training set. For this purpose we use the 8K dataset which described in Section 3.4.

Supervised machine learning methods have to be trained before they can be used for classification tasks. Therefore, we have to convert each document d into a document vector $\vec{d}$. These vectors use several different numerical features to represent documents. Each vector has length m where m is the number of distinct features in the training set. We will now give a short introduction to the three supervised learning methods we are using. All of them are implemented in the Python package `sklearn`[1].

4.2.1 Naive Bayes

Naive Bayes is the first supervised learning method we use to assign a class c to a given document d. We chose the class for which we retrieved the highest probability $c = \arg\max_c P(c|d)$. Using Bayes theorem,

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)} \tag{3}$$

we observe that $P(d)$ plays no role in choosing a class c. Each document d consists of different words, or *features* f_i, which must first be converted into a numerical

[1] http://scikit-learn.org, last lookup November 4, 2017

feature vector. $\langle f_1, f_2, ..., f_m \rangle$ are the features in d that are part of the vocabulary we use for the classification and m is the number of such tokens in d [26]. Naive Bayes assumes, that each feature is independent of other features. Hence, we maintain the Naive Bayes (NB) classifier:

$$P_{NB}(c|d) = \frac{P(c)(\prod_{i=1}^{m} P(f_i|c))}{P(d)} \tag{4}$$

The assumption that the occurrence of some words is independent of the appearance of other words is of course far from reality. However, Lewis [27] found that Naive Bayes performs surprisingly well on some problems.

4.2.2 Random Forest

Random Forest [28] uses, as the name suggests, a collection of random generated decision trees. During training, the algorithm choses random samples with replacement from the training set and builds different decision trees by choosing a random set of features at each splitting point. To classify documents, the algorithm traverses through the different trees, where each tree in the forest returns a class suggestion, or a vote, for each document. A document is classified based on the majority vote of all trees. This is repeated for each document in the collection. This procedure is displayed simplified in Figure 2.

4.2.3 Support Vector Machines

Support Vector Machines (SVM) [29] have proven to work very well on text classification, generally outperforming Naive Bayes [30]. In contrast to Naive Bayes and Random Forest, which are both probabilistic classifiers, SVM is a *large margin* classifier, since its goal is to find a decision boundary between (in our case) two classes. Therefore, during the training process (for the two-classes case), SVM aims

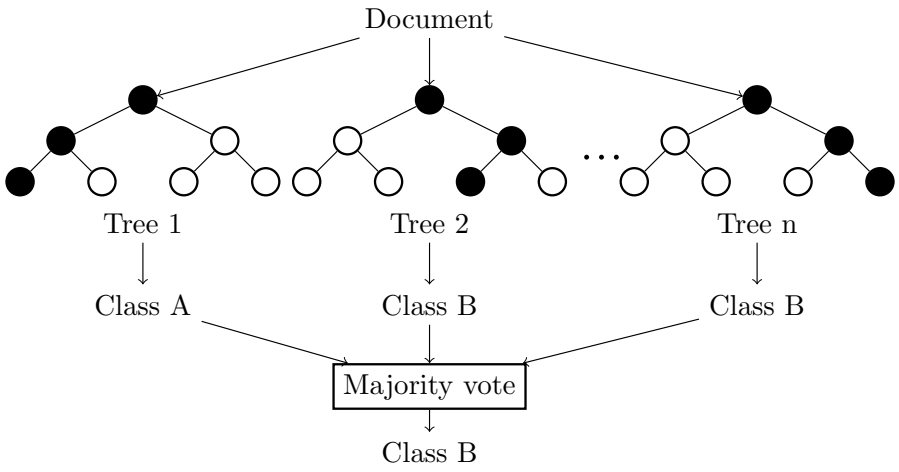

Figure 2: Random Forest scheme

to deploy a hyperplane, represented by a vector $\vec{w}$, that divides different classified document vectors from each other. Furthermore, this hyperplane should also maximize the separation, so that the *margin* between hyperplane and document vectors is as large as possible. Finding a proper hyperplane corresponds to a constrained optimization problem. Letting $c_i \in \{-1, 1\}$ be the class for a document vector $\vec{d_i}$, then the solution for the hyperplane can be described as follows:

$$\vec{w} := \sum_i \alpha_i c_i \vec{d_i}, \quad \alpha_i \geq 0 \tag{5}$$

Finding the α_i's is the aim of a dual optimization problem. All document vectors $\vec{d_i}$ that are located on the margin (such that $\alpha_i \neq 0$) are called *support vectors*. After the training procedure, a document is classified by determining on which side of the hyperplane the corresponding document vector is located.

4.3 Experimental Set-up

We use different kind of text processing methods combined with different scoring and weighting models. All combinations have the following three steps in common.

1. We prepare a document d as outlined in Section 4.1. After that, we calculate the sentiment value s_d for this text using equation (2).

2. We group all texts by the date on which they were posted (for the Financial Times we use the day on which the texts were published) and sum up all sentiment values for each trading day. Texts posted on non-trading days are shifted to the next trading day, as we assume that news on non-training days are still relevant for the next following trading day.

3. We then normalize the sentiment sum of a day by the number of texts on this day. If the sentiment sum on a day is *negative* we will predict the stock price on this day as *falling*. If the sum is *positive* or zero we will predict *rising*.

After the preprocessing procedure, we evaluate the effect of the following methods on our predictions.

Weighting models We use the following three models to weight the sentiment value s_d for each news document d. Each model either uses the number of up-votes v, the number of comments c or both. To weight an article we multiply the sentiment value with the result of a weighting function w. The weighting function uses a logarithmic function for smoothing purposes, since some submissions retrieved more than 10,000 up-votes, whereas others did not receive a single up-vote. To prevent submissions with a very small weight from getting ignored we establish a minimum weight w_{min} for each model. We optimize w_{min} for each model to maximize our accuracy.

The first model M1 uses the number of votes v to weight the relevance of an submitted article. This weighting function is described in the following equation:

$$w(v, w_{min}) = \max\{\log_2(v+1), w_{min}\} \tag{6}$$

The second model M2 uses the number of comments c for weighting the sentiments, as described as follows:

$$w(c, w_{min}) = \max\{\log_2(c+1), w_{min}\} \tag{7}$$

Our third model M3 uses both, the number of votes v and the number of comments c of an article to weight the calculated sentiment of this article. For this purpose we first multiply the sentiment score s_d with $w(v, w_{min1})$ and subsequently with $w(c, w_{min2})$. w_{min1} and w_{min2} have to be found through optimization.

Before we sum up all sentiments for a trading day we normalize all sentiments with the highest absolute sentiment value.

Scoring We introduce scoring to put emphasis on the relevance of the words in our sentiment dictionaries. As scoring method we use BM25 [31] to rank every word in a news document that is also part of our sentiment dictionary. Thus, scoring is not used in combination with machine learning. Before we are able to use BM25 for scoring we need to pre-calculate the inverse document frequency IDF(w_i) for each word w in a document collection (i.e. the Reddit news set and the Financial Times set). The BM25 score for a document d and the intersection W of words in d and in the sentiment dictionary is calculated as shown in equation (8).

$$score(W, d) = \sum_{w \in W} \text{IDF}(w) \cdot \frac{\text{tf}_d(w) \cdot (k+1)}{\text{tf}_d(w) + k \cdot (1 - b + b \cdot \frac{|d|}{\text{avgdl}})}, \tag{8}$$

where $\text{tf}_d(w)$ is w's term frequency in d, $|d|$ is the word count of d and avgdl is the average word count for all articles. The parameters k and b are tuning parameters, whereby k is normally set to a value between 1.2 and 2 and $b = 0.75$ [26] (we used $k = 1.75$; $b = 0.75$). Parallel to this, we calculate the sentiment value as ever, to determine whether the sentiment of an article is either positive or negative. We then replace the calculated sentiment value with the BM25 score for this news text, whereby we preserve the algebraic sign. Again, we use normalization with the highest absolute sentiment value (which in fact is now the BM25 score) before we sum up all sentiments on a day.

Noun phrases We use OpinionFinder [32] to extract noun phrases from our collected news texts. Quirk [33] defines a noun phrase as a subject, object and complement of a clause, and as a complement of a prepositional phrase and should capture a richer linguistic representation [34].

Stemming As the natural language contains different forms of a word due to grammatical reasons we also try to match word stems instead of whole words [26]. For this purpose we applied the `Porter 2` [35] stemming algorithm (implemented in the Python package `stemming`) on both the sentiment dictionary and our collected texts.

Doc2Vec Doc2Vec [36] is another technique to represent texts as vectors. It is based on Word2Vec [37]. For example, the words *Paris*, *France* and *power*. In a bag-of-words based representation they all would have the same relation to

each other. Though, we would consider *Paris* to be closer to *France* than to *power*. Word2Vec preserves these kind of relations. We will evaluate, whether this representation enhances the accuracy of our predictions. For this purpose we used the implementation of the Python package `gensim`[2].

[2]https://radimrehurek.com/gensim/index.html, last lookup November 5, 2017

5 Results

This chapter presents the results we achieved with our different models and compares the accuracy of them. The accuracy of a prediction describes on how many days a model predicted the right direction of the stock index.

Due to the missing time we were not able to perform an appropriate tuning of the hyper-parameters for the machine learning algorithms. This is normally a very time consuming and complex task, depending on the amount of parameters and possible values. Nevertheless, we want to describe a technique that can be used for hyper-parameter tuning, namely grid search. In detail, grid search tries to find the best parameters by testing all specified values for all specified parameters. This evaluation is based on a training and a test set. In our case we would divide the 8k news set into two subsets, where the test set contains the last 10% (referred to the date) of the news in the whole set and the training set contains the rest. These hyper-parameters for which the accuracy on the test set is maximised, are considered to be the best for this specific setting. However, we tried to find reasonable parameters for our models, which are displayed in Table III.

The accuracy of the naive baseline, which in fact is random guessing based on the majority class, is 54.2%. Thus, following our results displayed in Table IV and Table V, we see that each of our models outperforms this naive baseline. Only the models based on a Doc2Vec representation perform worse. Furthermore, most of our models outperform the unweighted predictions based on Financial Times

Table III: The hyper-parameters we are using for our three machine learning algorithms (note: `sklearn` version 0.19.1). The remaining parameters use default values.

Naive Bayes	alpha=0
Support Vector Machines	C=10, kernel="rbf"
Random Forests	n_estimators=130, oob_score=True, min_samples_leaf=100, criterion="gini"

articles, which are approximately as accurate as random guessing.

The bag-of-words based approaches performed surprisingly well, even though splitting a text into single words brings some weaknesses as stated by Le and Mikolov [36]. They stressed that representing a text as a list of words is followed by a loss of word ordering and an ignorance of the semantics of the words. To counter these weaknesses we tested whether the use of noun phrases is a more suitable way to represent the news texts in our collection. Our results show clearly that the use of nouns do indeed increase the accuracy of our predictions. In case of Naive Bayes and model M2 we achieved an enhancement of the accuracy by 1.96. Indeed, we achieved the best results by using nouns. This holds for the dictionary-based models as well as for machine learning. With Naive Bayes using model M3 we achieved 56.49%. Looking at the results shown in Table IV we see that using noun phrases increases the accuracy for our dictionary-based approaches as well. The best result for this approach are the models M2 and M3, with 55.56% accuracy. However, only for the unweighted model there is a decrease of 0.14.

Using Doc2Vec to represent texts should maintain the natural relations between words rather more than bag-of-words based models. Although, we observe that this did not enhance the accuracy of our models. Only with Random Forest we achieved accuracies of over 50%.

We notice that the use of stemming affects the accuracy rather in a negative

than a positive way. In comparison with the use of nouns the accuracy of the non-scoring models M1-M3 decreased by approximately 1; at most 1.13 for model M2. We used stemming with the aim to achieve more possible matches in the news texts. Annett and Kondrak [38] evaluated different techniques to determine the sentiment of textual data and came to the a similar conclusion respective stemming. They assumed that the additional word matches are counter-balanced by the loss of information represented by morphological endings.

We implemented scoring with the intention to rank documents depending on the term and inverse document frequency of the matches in this document. Though, this has rather a slightly negativ effect on the models than a positive effect. Especially model M3, with a decrease of 0.92 for the bag-of-words approach. However, only the accuracy of the unweighted models (except for stemming) profits from using scoring.

We derived weighting models for a submission using its number of votes and number of comments. After applying weighting, we observe that this does not increase the accuracy in all cases. Though, for some models we achieved an enhancement of more than 1. It seams that especially the accuracy of noun phrases profit from weighting (e.g. Naive Bayes with an increase of 1.25 compared to the unweighted model). However, in some cases weighting does not affect the accuracy at all or even in a negative manner.

Table IV: Accuracy (as a percentage) on predictions using dictionary-based approaches

			Reddit			
	Scoring	Financial Times	Unweighted	M1	M2	M3
Bag of words	None	54.19	54.59	54.68	54.80	54.76
	BM25	54.83	54.76	54.30	54.26	53.84
Stems	None	54.20	54.39	54.47	54.43	54.55
	BM25	54.20	54.30	54.30	54.30	53.84
Noun phrases	None	54.24	54.45	55.45	**55.56**	**55.56**
	BM25	54.02	54.83	54.41	54.52	54.41

Table V: Accuracy (as a percentage) on predictions using machine learning

			Reddit			
	Feature representation	Financial Times	Unweighted	M1	M2	M3
Naive Bayes	Bag of words	54.20	54.85	54.43	54.43	54.72
	Nouns	54.24	55.24	55.45	**56.39**	**56.49**
	Doc2Vec	45.80	46.27	46.48	46.23	46.31
Support Vector Machines	Bag of words	54.20	54.51	54.42	54.43	54.43
	Nouns	54.24	55.45	55.45	55.45	55.45
	Doc2Vec	47.89	50.50	51.13	51.93	51.63
Random Forest	Bag of words	54.20	54.85	55.02	54.89	54.97
	Nouns	54.24	55.42	55.45	55.45	55.56
	Doc2Vec	54.20	53.86	53.77	54.07	53.52

6 Discussion

The results produced by our dictionary-based approaches are quite good in comparison to the machine learning techniques. Though, machine learning performs generally better. Furthermore, we observe that most of the models we developed for this thesis outperform the naive baseline. Using noun phrases as well as using a stemming algorithms affects the results. We see that splitting a text into its nouns does generally tend to lead to a higher accuracy. In fact, this holds for both approaches, machine learning and the dictionary-based classification. However, the use of stemming does only slightly affect the accuracy but most of all in a negative manner. Thus, we argue that stemming has no use for our intention. The use of weighting functions, however, is able to enhance the accuracy for some models, especially when applied on noun phrases. Although, weighting did also decrease the accuracy of some models. Thus, it is quite difficult to evaluate the use of weighting. However, we find that using votes and comments is reasonable when using the right weighting functions. In fact, we achieved the best result with an accuracy of 56.49% using Naive Bayes on noun phrases weighted with both, votes and comments.

The machine learning techniques we are using do generally outperform the dictionary-based approaches in many cases. In detail, Random Forests and Naive Bayes perform approximately equally well. Support Vector Machines showed some disadvantages in our experiment when applying it on bag-of-words feature vectors. On the other hand, applied on noun phrases it achieved a similar accuracy as

Random Forests did. However, a disadvantage of the machine learning approach is the time it takes to train its models. In fact, this problem grows with a larger data set.

During the evaluation of the sentiments in our news text collection we noticed that our approaches classify many more texts as negative rather than positive. The unweighted model classifies only 7,546 news as positive and 35,659 as negative while using the sentiment dictionary. Whereas, the Naive Bayes classifier assigns 7,777 news to positive and 35,205 news to negative. Hence, tried to evaluate reasons for this negativity bias. We assume that one reason could be that there are more negative than positive words in the sentiment dictionary. Furthermore, we consider the bag-of-words representation and, hence, the loss of contextual information [39] to be another possible reason for this bias. Therefore, we argue that the loss of context and semantics could lead to a misinterpretation of a texts relating to its tone. For example, if we consider the word *abuse*. Standing on its own one would most certainly classify it as negative. But if we imagine a context in which someone was able to stop some kind of *abuse* the meaning of a text would be considered to be positive now. Even though, a dictionary-based model could still classify this text as negative, since it contains the word *abuse*. This is clearly a disadvantage of single-word dictionaries since they are not able to image the behaviour of natural language appropriate.

Another explanation for our observation could be that negative news are generally much more frequent than positive news. Soroka and McAdams [40] tried to find reasons for the assumption that there is a certain negativity bias in political and economical news. They examined the psychophysiological behaviour of human beings and found that humans pay more attention to negative information, which is why there is such an emphasis on negative news on mass media. In a reverse conclusion this means that many positive reports are considered either a

unimportant by the media and hence, not published or that companies did not even distribute them in the first place, following the phrase "no news is good news". However, this finding naturally also affects the news posted on Reddit.

During testing we identified some problems and bottlenecks. The first bottleneck of our approach is the dynamic of Reddit itself. We remember that submissions that gain a lot of attention by the users can stay in the *hot* section for days. In this time, these submissions will most certainly receive even more votes and comments. This makes it difficult to use Reddit as a source for real-time predictions, since it takes some time for the number of comments and votes to stabilize. Another challenge is downloading the news posted on Reddit. As described previously we have to provide and maintain rules for our crawler so it is able to retrieve the texts from the respective website. Of course, a solution in which the crawler downloads all paragraph HTML tags would be possible. Although, this will lead to impure data since there exist no rules for website administrators how to format texts on their websites.

7 Conclusion

We have investigated the relation between S&P 500 closing values and the sentiment of news texts obtained through the social news aggregator Reddit. We used a dictionary-based approach as well as supervised machine learning techniques to analyze the tone of these news. After that, we developed certain models that involve the votes and comments to weight the relevance of submissions. The results we achieved based on the sentiment analysis do indeed outperform the naive baseline as well as the results based on the Financial Times slightly. Therefore, we state that Reddit has a certain power to predict stock index movements. Furthermore, our results show that the number of votes and comments are reasonable measures to weight the relevance of news articles.

We find that Reddit has a certain potential to be used as a source for stock market predictions. A topic for future work could be to develop and maintain a real-time system and to create trading strategies from it. This would also include to extend our model in a way that it predicts real values. A last thing could be to evaluate models that include the content of comments into their predictions.

8 Bibliography

[1] B. G. Malkiel and E. F. Fama, "Efficient capital markets: A review of theory and empirical work", *The journal of finance*, vol. 25, no. 2, pp. 383–417, 1970.

[2] G. Gidofalvi and C. Elkan, "Using news articles to predict stock price movements", *Department of computer science and engineering, university of california, san diego*, 2001.

[3] A. Mittal and A. Goel, "Stock prediction using twitter sentiment analysis", *Standford university, cs229*, vol. 15, 2012.

[4] K. C. Butler and S. J. Malaikah, "Efficiency and inefficiency in thinly traded stock markets: Kuwait and saudi arabia", *Journal of banking & finance*, vol. 16, no. 1, pp. 197–210, 1992.

[5] M. G. Kavussanos and E. Dockery, "A multivariate test for stock market efficiency: The case of ase", *Applied financial economics*, vol. 11, no. 5, pp. 573–579, 2001.

[6] J. D. Thomas and K. Sycara, "Integrating genetic algorithms and text learning for financial prediction", *Data mining with evolutionary algorithms*, pp. 72–75, 2000.

[7] L. Blume, D. Easley, and M. O'hara, "Market statistics and technical analysis: The role of volume", *The journal of finance*, vol. 49, no. 1, pp. 153–181, 1994.

[8] M. J. Pring, *Technical analysis explained: The successful investor's guide to spotting investment trends and turning points*. McGraw-Hill Professional, 2002.

[9] X. Li, H. Xie, L. Chen, J. Wang, and X. Deng, "News impact on stock price return via sentiment analysis", *Knowledge-based systems*, vol. 69, pp. 14–23, 2014.

[10] M. Cecchini, H. Aytug, G. J. Koehler, and P. Pathak, "Making words work: Using financial text as a predictor of financial events", *Decision support systems*, vol. 50, no. 1, pp. 164–175, 2010.

[11] V. Niederhoffer, "The analysis of world events and stock prices", *The journal of business*, vol. 44, no. 2, pp. 193–219, 1971.

[12] B. Wüthrich, D Permunetilleke, S. Leung, W Lam, V. Cho, and J Zhang, "Daily prediction of major stock indices from textual www data", *Hkie transactions*, vol. 5, no. 3, pp. 151–156, 1998.

[13] R. P. Schumaker and H. Chen, "Textual analysis of stock market prediction using breaking financial news: The azfin text system", *Acm transactions on information systems (tois)*, vol. 27, no. 2, p. 12, 2009.

[14] M. Ammann, R. Frey, and M. Verhofen, "Do newspaper articles predict aggregate stock returns?", *Journal of behavioral finance*, vol. 15, no. 3, pp. 195–213, 2014.

[15] R. J. Dolan, "Emotion, cognition, and behavior", *Science*, vol. 298, no. 5596, pp. 1191–1194, 2002.

[16] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk", *Econometrica: Journal of the econometric society*, pp. 263–291, 1979.

[17] J. R. Nofsinger, "Social mood and financial economics", *The journal of behavioral finance*, vol. 6, no. 3, pp. 144–160, 2005.

[18] J. Bollen and H. Mao, "Twitter mood as a stock market predictor", *Computer*, vol. 44, no. 10, pp. 91–94, 2011.

[19] Reddit. (Sep. 8, 2017). Audience and demographic, [Online]. Available: `https://www.redditthelp.com/en/categories/advertising/advertising-101/audience-and-demographic`.

[20] B. D. Horne and S. Adali, "The impact of crowds on news engagement: A reddit case study", *Arxiv preprint arxiv:1703.10570*, 2017.

[21] E. Gilbert, "Widespread underprovision on reddit", in *Proceedings of the 2013 conference on computer supported cooperative work*, ACM, 2013, pp. 803–808.

[22] C. Buntain and J. Golbeck, "Identifying social roles in reddit using network structure", in *Proceedings of the 23rd international conference on world wide web*, ACM, 2014, pp. 615–620.

[23] C. Liu, J. Wang, D. Xiao, and Q. Liang, "Forecasting s&p 500 stock index using statistical learning models", *Open journal of statistics*, vol. 6, no. 06, p. 1067, 2016.

[24] T. Loughran and B. McDonald, "When is a liability not a liability? textual analysis, dictionaries, and 10-ks", *The journal of finance*, vol. 66, no. 1, pp. 35–65, 2011.

[25] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques", in *Proceedings of the acl-02 conference on empirical methods in natural language processing-volume 10*, Association for Computational Linguistics, 2002, pp. 79–86.

[26] D. M. Christopher, R. Prabhakar, and S. Hinrich, "Introduction to information retrieval", in *Proceedings of the international communication of association for computing machinery conference*, 2008.

[27] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval", in *European conference on machine learning*, Springer, 1998, pp. 4–15.

[28] L. Breiman, "Random forests", *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[29] C. Cortes and V. Vapnik, "Support-vector networks", *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[30] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features", *Machine learning: Ecml-98*, pp. 137–142, 1998.

[31] S. Robertson, H. Zaragoza, *et al.*, "The probabilistic relevance framework: Bm25 and beyond", *Foundations and trends® in information retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[32] T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan, "Opinionfinder: A system for subjectivity analysis", in *Proceedings of hlt/emnlp on interactive demonstrations*, Association for Computational Linguistics, 2005, pp. 34–35.

[33] R. Quirk, *A comprehensive grammar of the english language*. Pearson Education India, 2010.

[34] P. G. Anick and S. Vaithyanathan, "Exploiting clustering and phrases for context-based information retrieval", in *Acm sigir forum*, ACM, vol. 31, 1997, pp. 314–323.

[35] M. F. Porter, “An algorithm for suffix stripping”, *Program*, vol. 14, no. 3, pp. 130–137, 1980.

[36] Q. Le and T. Mikolov, “Distributed representations of sentences and documents”, in *Proceedings of the 31st international conference on machine learning (icml-14)*, 2014, pp. 1188–1196.

[37] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space”, *Arxiv preprint arxiv:1301.3781*, 2013.

[38] M. Annett and G. Kondrak, “A comparison of sentiment analysis techniques: Polarizing movie blogs”, *Advances in artificial intelligence*, pp. 25–35, 2008.

[39] T. Li, T. Mei, I.-S. Kweon, and X.-S. Hua, “Contextual bag-of-words for visual categorization”, *Ieee transactions on circuits and systems for video technology*, vol. 21, no. 4, pp. 381–392, 2011.

[40] S. Soroka and S. McAdams, “News, politics, and negativity”, *Political communication*, vol. 32, no. 1, pp. 1–22, 2015.