

Mestrado em Gestão de Informação
Master Program in Information Management

**APPLYING TEXT MINING TECHNIQUES TO FORECAST
THE STOCK MARKET FLUCTUATIONS OF LARGE IT
COMPANIES WITH TWITTER DATA**

Descriptive and predictive approaches to enhance
the research of stock market predictions with textual
and semantic data

Christos Zois

Project proposal presented as partial requirement for
obtaining the Master's degree in Information Management

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

**APPLYING TEXT MINING TECHNIQUES TO FORECAST THE STOCK
MARKET FLUCTUATIONS OF LARGE IT COMPANIES WITH TWITTER
DATA**

Descriptive and predictive approaches to enhance the research of
stock market predictions with textual and semantic data

By Christos Zois

Project proposal presented as partial requirement for obtaining the Master's degree in Information Management, with a specialization in Information Systems and Technologies Management

Advisor / Co Advisor: *Professor Doctor Roberto Henriques*

Co Advisor: *Professor Doctor Mauro Castelli*

ABSTRACT

This research project applies advanced text mining techniques as a method to predict stock market fluctuations by merging published tweets and daily stock market prices for a set of American Information Technology companies. This project executes a systematical approach to investigate and further analyze, by using mainly R code, two main objectives: i) which are the descriptive criteria, patterns, and variables, which are correlated with the stock fluctuation and ii) does the single usage of tweets indicate moderate signal to predict with high accuracy the stock market fluctuations. The main supposition and expected output of the research work is to deliver findings about the twitter text significance and predictability power to indicate the importance of social media content in terms of stock market fluctuations by using descriptive and predictive data mining approaches, as natural language processing, topic modelling, sentiment analysis and binary classification with neural networks.

KEYWORDS

Text Mining; Data Mining; Predictive Model; Topic Modelling; Stock Market; Social Media Analysis;
Binary Classification

INDEX

- 1. Introduction.....1
 - 1.1. Research context and Problem Identification2
 - 1.2. Definition of research questions and project goal3
- 2. Study relevance and importance.....4
 - 2.1. Project objectives4
 - 2.2. Research Methodology.....5
- 3. Literature review7
 - 3.1. Fundamentals of data mining.....7
 - 3.1.1. Data structures and data types8
 - 3.1.2. Data mining and Machine Learning.....9
 - 3.2. Predictive modeling and Classification algorithms10
 - 3.2.1. Classification and Regression Trees.....10
 - 3.2.2. Logistic Regression.....11
 - 3.2.3. Neural Networks.....11
 - 3.3. Fundamentals of text mining.....12
 - 3.3.1. Data preprocessing in text mining.....12
 - 3.3.2. Sentiment analysis.....14
 - 3.3.3. Latent Dirichlet Allocation15
 - 3.3.4. Limitations with text mining.....16
 - 3.4. Stock market and predictive capabilities17
 - 3.5. Reference projects in text mining for stock predictions18
 - 3.6. Applied R Libraries.....19
- 4. Project Methodology and Execution20
 - 4.1. Data Preprocessing.....21
 - 4.1.1. Extract and Preprocess the Data into R Studio21
 - 4.1.2. Explore Data Issues and apply Data Cleaning.....21
 - 4.1.3. Preprocessed Data Set.....24
 - 4.2. Data exploration24
 - 4.2.1. Examination of continuous variables25
 - 4.2.2. Examination of Outliers26
 - 4.2.3. Examination of categorical variables.....28
 - 4.2.4. Exploration of the text variable.....33
 - 4.2.5. Observations and resulting Actions.....38

4.3. Data Modification and Feature Engineering	39
4.3.1. Modification and Transformation of the Data Set	39
4.3.2. Final exploration and modification of the data set	47
4.3.3. Extracted Features and final Data Set	48
4.4. Machine Learning Classification	49
4.4.1. Model Selection for Binary Classification Problems.....	49
4.4.2. Data Partition.....	51
4.4.3. Feature Selection.....	52
4.4.4. Binary Classification with Neural Networks	56
4.4.5. Model Evaluation.....	59
5. Conclusion	62
5.1. Answer to the research questions.....	63
5.2. Limitations and future work.....	65
6. Bibliography.....	67
7. Annex.....	72

LIST OF FIGURES

- Figure 1 – Elon Musks tweet on Teslas bankruptcy 2
- Figure 2 – Graphical representation of research methodology and project structure 6
- Figure 3 –Graphical representation of the CRISP model..... 8
- Figure 4 – Phenomenon of underfitting and overfitting..... 9
- Figure 5 – Illustration of a decision tree..... 10
- Figure 6 – Illustration of a logistic regression 11
- Figure 7 – Representation of Neural Networks..... 12
- Figure 8 – Stages of text mining stemming techniques 13
- Figure 9 – Dirichlet Distributions after first iteration..... 16
- Figure 10 – Dirichlet Distributions after multiple iterations 16
- Figure 11 – Methodology and structure of present project work 20
- Figure 12 –Data model for the present project 22
- Figure 13 – Pearson correlation before the data modification..... 26
- Figure 14 – Boxplots indicate outliers 27
- Figure 15 – Boxplots for length of tweets 27
- Figure 16 – Outliers with heavy content 27
- Figure 17 – Wordcloud of threshold outliers 28
- Figure 18 – Screenshot of largest tweet with over 9.000 characters 28
- Figure 19 – Distribution of variable Increase based on company..... 30
- Figure 20 – Count of variable increase..... 30
- Figure 21 – Distribution of variable Increase based on length of the tweets..... 30
- Figure 22 – Distribution of Increase through date for Oracle..... 31
- Figure 23 - Distribution of Increase through date for Amazon 31
- Figure 24 - Distribution of Increase through date for Facebook..... 31
- Figure 25 - Distribution of Increase through date for Apple..... 31
- Figure 26 - Distribution of Increase through date for Google..... 31
- Figure 27 - Distribution of Increase through date for Microsoft 31
- Figure 28 – Oracle tweets frequencies for each hour per day..... 32
- Figure 29 – Amazon tweets frequencies for each hour per day 32
- Figure 30 – Facebook tweets frequencies for each hour per day..... 32
- Figure 31 – Apple tweets frequencies for each hour per day..... 32
- Figure 32 – Google tweets frequencies for each hour per day..... 32
- Figure 33 – Microsoft tweets frequencies for each hour per day 32
- Figure 34 – Tweet frequencies for each hour per day..... 33

Figure 35 – Count of increased values by trading hours.....	33
Figure 36 – Word cloud for the final data frame	34
Figure 37 – Frequency table of most frequent words.....	34
Figure 38 – Distribution of Duplicates within companies	34
Figure 39 – Distribution of duplicates.....	34
Figure 40 – LDA and top features Results for Oracle	35
Figure 41 – LDA and top features results for Microsoft.....	35
Figure 42 – LDA and top features results for Apple.....	35
Figure 43 – LDA and top features results for Google.....	35
Figure 44 – LDA and top features results for Amazon	35
Figure 45 – LDA and top features results for Facebook.....	35
Figure 46 – Top 10 words of Oracle	36
Figure 47 – Top 10 words of Microsoft	36
Figure 48 – Top 10 words of Apple.....	36
Figure 49 – Top 10 words of Google	36
Figure 50 – Top 10 words of Amazon.....	36
Figure 51 – Top 10 words of Facebook	36
Figure 52 – Word-topic probabilities for Oracle	37
Figure 53 – Word-topic probabilities for Microsoft	37
Figure 54 – Word-topic probabilities for Google	37
Figure 55 – Word-topic probabilities for Apple	37
Figure 56 – Word-topic probabilities for Amazon.....	37
Figure 57 – Word-topic probabilities for Facebook	37
Figure 58 – Polarity histogram	39
Figure 59 – Proportion of increased values by subsetting trading hours	40
Figure 60 – Proportion of increased values by subsetting companies	40
Figure 61 – Density of sentiment analysis with the package qdap.....	44
Figure 62 – Boxplots of sentiment analysis with the package qdap.....	44
Figure 63 – Polarity histogram for each company	44
Figure 64 – Sentiments with nrc.....	46
Figure 65 – Sentiments with bing.....	46
Figure 66 – sentiments with afinn.....	46
Figure 67 – Positive words in unique tweet values.....	48
Figure 68 – Negative words in unique tweet values.....	48
Figure 69 – Top frequent positive terms.....	48
Figure 70 – Top frequent negative terms.....	48

Figure 71 – Pearson Correlation for continuous variables..... 53
Figure 72 – Spearman correlation for continuous variables..... 53
Figure 73 – Variable Importance results of Neural Network..... 55
Figure 74 – Scatterplot Matrix for the used features 56
Figure 75 – Relative Variable Importance for final Neural Network..... 60
Figure 76 – ROC Curve for NNL4 60

LIST OF TABLES

Table 1 – Selection of applied R libraries 19

Table 2 – Variables of first data set..... 24

Table 3 - Univariate analysis of the first data set..... 25

Table 4 – Overview of descriptive statistics for the factor variables..... 28

Table 5 – Summary of observations and associated modification actions..... 38

Table 6 – Final data set after the second cleaning phase for all companies with a set of 60 variables 49

Table 7 – Cost-Benefit Analysis for selected machine learning models 51

Table 8 – Results of Chi-square test..... 52

Table 9 – Variable importance results of Logistic Regression..... 54

Table 10 – Variable importance results of CART..... 55

Table 11 – Results of Neural Networks 58

Table 12 – Model selection for 1 hidden layer with sample=3000..... 60

Table 13 – Feature Selection Iterations with NNL4 model 61

Table 14 – Data sample Iterations with NNL4 model 61

Table 15 – Final data set..... 72

LIST OF SCRIPTS

- Script 1 – Split the variable created_at 22
- Script 2 – Deleting NA and cleaning the text variable..... 23
- Script 3 – Merging the twitter and yahoo data set 23
- Script 4 – Calculating Pearsons Correlation 26
- Script 5 – Building a boxplot to detect outliers..... 26
- Script 6 – Building word clouds to examine the content of the outliers 27
- Script 7 – Exploring the target variable 29
- Script 8 – Investigating the distribution of the tweet lengths 30
- Script 9 – Performing drill-downs to examine the target variable..... 31
- Script 10 – Building word clouds and frequency tables to examine word frequencies..... 33
- Script 11 – Performing LDA to output the top 10 topics..... 35
- Script 12 – Building based on the LDA result the word-probability plots..... 36
- Script 13 – Building the variable stock_content by setting up a lexicon 38
- Script 14 – Illustrating the distribution of the target variable within the trading hours 40
- Script 15 – Classifying unique tweets by training a classification tree 41
- Script 16 – Performing and merging the sentiment analysis output with the existing data set
..... 43
- Script 17 – Applying tidy sentiment analysis on the data set 45
- Script 18 – Unnest the vectors and extending the variable by transforming them into integers
..... 47
- Script 19 – Classifying the polarities in numeric sentiments 47
- Script 20 – Creating a data sample for training and test data set 52
- Script 21 – Applying Logistic Regression to analyze variable importance 54
- Script 22 – Applying Classification and Regression Trees to analyze variable importance 54
- Script 23 – Applying Neural Networks to analyze variable importance 54
- Script 24 – Converting and normalizing the variables 56
- Script 25 – Training the first neural network 58
- Script 26 – Calculating model selection metrics 59

LIST OF ACRONYMS

ACRONYM	DESCRIPTION
AIC	Akaike Information Criterion
API	Application Programming Interface
AUC	Area under the Curve
BIC	Bayesian Information Criterion
CART	Classification and Regression Trees
CRISP	Cross Industry Standard Process for Data Mining
CSV	Comma-separated-value
ETL	Extract, Transform and Load
IT	Information Technology
LDA	Latent Dirichlet Allocation
NLP	Natural Language Processing
NNL	Neural Networks
ROC	Receiver Operating Characteristics

1. INTRODUCTION

With the introduction and commercialization of the internet in the 1990s, the velocity of reactions and opinions for certain news started to increase drastically. Indeed, the present society spreads information in velocity and variety, which impacts the economical behavior radically. This is possible due to the ubiquity of digital news platforms and the resulting responses on social media channels such as tweets. Twitter is seen as one of the largest opinion-based networks of the internet, generating daily hexabytes of data, which influences globally the purchasing behavior and fluctuates the stock market price (Bollen, Mao, & Zeng, 2016).

Simultaneously the development of computing was disrupting the economical field, in the fields of data mining, which optimizes decision making processes and firm strategies (Giudici & Figini, 2009). Particularly, the opportunity to apply data mining methods on the generated big data from social media channels is highly valuable and is applied ever since for financial market predictions (Beckmann, 2017). In fact, applying data mining on text structures, also known as text mining, can be a great supporting asset, especially for decision-making processes. Taking in consideration the major number of text generated in twitter, data mining proposes machine learning algorithms, which can describe and predict with high accuracy economical fluctuations and help organizations to rebuild social media channels, as well as to forecast their daily outcome in the stock market (Bollen, Mao, & Zeng, 2016). Further, Bollen et al describe the value of applying text mining techniques for financial market descriptions. In detail, they describe how sentiment analysis can support the prediction of fluctuations in the market, and how businesses can use this knowledge to restructure radically their decision-making processes.

Now the question arises, how the collection of multiple opinions, which de facto the output of twitter is, can be used for future predictions. Technically speaking, every tweet contains information of the sender, the text, as well as reaction on it, the addressed topic, the addressee, inclusive a specific timestamp. Using the time stamp as a foreign key, every unique tweet can be matched with another database, for example with stock market data sets. In combination with external resources, patterns can be discovered, and this results in conclusions about how social media affect economic behavior.

The Information Management School of NOVA University Lisbon (NOVA IMS) understands the importance of this phenomenon and believe in knowledge discovery approaches and initiates, together with the Master Student Christos Zois, a working group which will apply text mining techniques with tweets collected for a certain time period for multiple technology-oriented

companies, namely Amazon, Apple, Google, Facebook, Microsoft and Oracle. As an outcome, the analysis will result in a selection of variables with high importance, as well as predictions for stock market values. The following chapters summarize the project procedure, describe the motivation and objectives of the project and explain the variety of work packages.

1.1. RESEARCH CONTEXT AND PROBLEM IDENTIFICATION

The correlation between tweets and stock markets can be illustrated with one prominent example from the spring of 2018, when the founder of Tesla, Elon Musk, published on the 1st April 2018 an ironic tweet about his bankruptcy for public relations purposes. Originally published with entertaining intention, but not representing the reality, this tweet negatively affected the stock value for Tesla for a measured 31-day period by negative 22% downgrade with a relatively low retweet quote of almost 25 thousand people (Russia Today, 2018). The effect on the stock market is illustrated below:

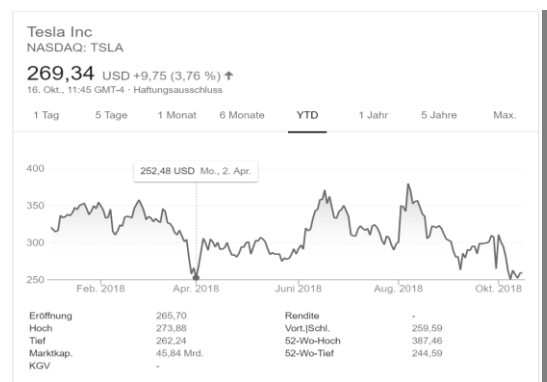


Figure 1 – Elon Musks tweet on Teslas bankruptcy¹

Indeed, recent studies revealed high correlations between several environmental, socio-demographical, and economic factors and stock market fluctuations (Nikfarjam, Emadzadeh, & Muthaiyah, 2018) (Tsai & Hisao, 2010) (Jadhav & Wakode, 2017). Translating this assumption for data mining purposes, it means that both qualitative and quantitative measurements are necessary to approach precise stock market predictions. On the other hand, many attempts to analyze historic stock data with established machine learning algorithms were not able to justify the random behavior of the stock market. Nikfarjam et al explain that *“the impact of un-quantifiable events on the market”* (Nikfarjam, Emadzadeh, & Muthaiyah, 2018) is absolutely necessary to enhance accurate predictive models.

¹ Copied from Google (2018)

Addressing Beckman's studies, social media has a great influence on the stock market fluctuations (Beckmann, 2017). Additionally, investigated data mining studies included text mining activities in their research to build prognosis capabilities on the stock market fluctuation. Nikfarjam et al define this activity as “news-based stock market prediction systems” (Nikfarjam, Emadzadeh, & Muthaiyah, 2018). This activity has the goal to classify whether the news content in the published time reached a positive or negative difference compared to the past stock value. Furthermore, their study includes a comparison of developed stock market classification systems for the identification of common dominators and features for text mining in stock market predictions. The findings illustrated that many data scientists and statisticians use sentiment analysis and semantic approaches to build predictive models, having as input data the news content and as a target the classification on positive or negative effects.²

It is relevant to investigate the social media activities of big companies to identify, how specific words, comments, opinions, and segments can have great impact on the fluctuation in the stock market, and hence, to split unstructured data into more valuable means for prediction. For this purpose, historic twitter data from the twitter application platform interface (API) was extracted to investigate, how prone the stock market value reacts to different tweet sentiments.

1.2. DEFINITION OF RESEARCH QUESTIONS AND PROJECT GOAL

Addressing the previous observations, one goal is to contribute to the study area “News based Stock Market Predictions” and text mining, together with statistical insights based on the used features and keywords, which influence the stock market evolvement. This project will match tweets with stock market data. The stock market data will be extracted and preprocessed from the Yahoo Finance API. The tweets contain roughly 2 million observation points from various information technology companies, namely Amazon, Apple, Google, Facebook, Microsoft, and Oracle. Further, this project will focus on the following research questions to fulfil the project scope, which is to preprocess and clean the data set, to extract and modify features, as well as to conduct predictions, to apply experiments, to see if it is possible to predict accurately the stock market prediction by using only qualitative twitter data. The following research questions were defined:

Research question 1: What are the characteristics of the data set? How can descriptive approaches and features engineering support the model development, and is it possible to build a statistical signal for the model?

Research question 2: Which features, and keywords show influence on the stock market fluctuation?

² Many participants included micro and macro economical variables into their models (Tsai & Hisao, 2010)

2. STUDY RELEVANCE AND IMPORTANCE

The previous facts showed, that news-based and semantic content is highly relevant variables for accurate stock market predictions. As shown, the influence of news and social media are one of many qualitative variables, which will help to better forecast the, as Nikfarjam et al describe, “random behavior of the stock market”. An article from Harvard Business School further explains: “[the] future, like any complex problem, has far too many variables to be predicted” (Stibel, 2009). Stibel concludes that there will be never a machine learning model constructed, which quantifies all relevant variables for this equation.

Nonetheless, a study of the Technical University in Chicago reminds of the original propose of machine learning algorithms, which is to generalize its predictive capabilities from historic data, as well as to translate a problem into a mathematical equation (McAllester, Srebro, & Urtasun, 2006). Fundamentally, the value of every machine learning algorithm is to generalize the parameters from observations. To optimize them, the machine learning algorithm must continuously be improved in increments or so-called evolutions. This can happen through two different approaches, namely, update the parameter settings and/or include new historical data.³ Based on this motivation, the research on identifying semantic patterns within news content is highly attractive for stock prediction and must be analyzed in great detail. To achieve this, the research must include modern data mining approaches and synergize them with stock market knowledge. This work will align its research methodology with pioneers in the field of sentiment analysis and text mining, especially by using the project from Singh (2019) as a reference.

Several organizations will benefit from the provided insights; thus, it will help to build efficiently machine learning algorithms by providing a best practice from different sources and authors. Especially, data scientists can benefit from the feature selection phase, which indicates important insights, which features are relevant for the predictions. The script, which will be provided and explained throughout this paper, can be used by data scientists.

2.1. PROJECT OBJECTIVES

The main objective of this master thesis is to answer the research question by preprocess the delivered twitter data, apply text mining techniques to quantify the contents into statistical relevant features, apply descriptive analytics to describe the AS-IS state of the data, and to investigate, which keywords are strongly correlated with the stock market fluctuations. The following checklist

³ Reinforcement learning is conceptualizing this into one process

summarizes the objectives for the operational actions, which must be performed to provide a qualitative answer to the research questions:

- i) Summarize and review fundamental knowledge in the field of stock market prediction
- ii) Summarize and review fundamental knowledge in the field of text mining
- iii) Preprocess the twitter data and perform data cleaning tasks to reduce noise
- iv) Apply text mining techniques to quantify the unstructured text into relevant features
- v) Apply sentiment analysis to the data set to label their semantic behavior
- vi) Apply feature engineering, especially by indicating word-frequencies and relevancy
- vii) Predict the stock market fluctuation by targeting the variable *increase*⁴
- viii) Analyze which keywords were highly relevant to classify the prediction

2.2. RESEARCH METHODOLOGY

To give an answer and a tailor-made solution for the above-described study objectives and problems, this research paper plans to structure the scientific paper in nine parts, which are represented in the following work breakdown structure. Each box describes actions with the corresponding categorized tasks. The Output column is representing the value of the work, which is contributing towards predicting with high accuracy. In summary, the figure describes, that a detailed literature review must be executed to build business context and knowledge about the programming language R. Then it is necessary to preprocess the data with cleaning procedures to apply text mining operations. By that new features will be build, which will be analyzed with feature selection methods. All this will lead then to the point, to use a selected machine learning algorithm to predict the present binary classification problem.

⁴ In this context, increase is the binary target variable, which indicates if the stock price rised or dropped in comparison to the last day.

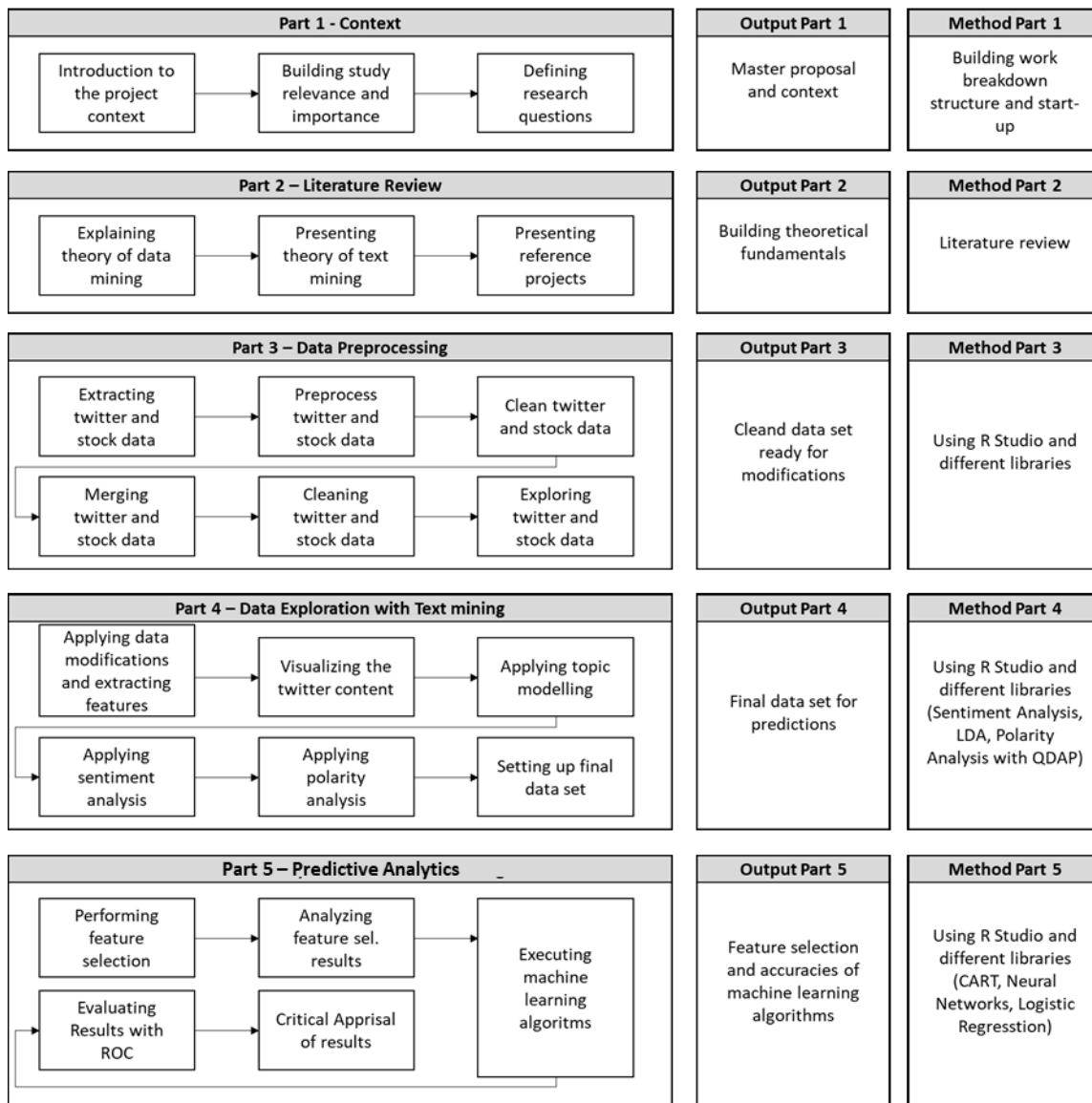


Figure 2 – Graphical representation of research methodology and project structure

3. LITERATURE REVIEW

This chapter discusses the results of a literature review in the field of text mining approaches for stock market prediction to provide the reader with fundamental knowledge in the area of news based predictive modeling, as well as to assure that the present project will produce original ideas, methods, and technical contents.⁵ Four areas of research are radically influencing the present project success, namely data mining, text mining, stock market predictions and the R programming language for the operational work. The results are consolidated in the following segments, namely fundamentals of data mining, fundamentals of text mining, machine learning algorithms, stock market predictive capabilities, reference projects in text mining for stock market predictions, as well as resulting actions for the project execution.

3.1. FUNDAMENTALS OF DATA MINING

Data mining is a bundle of statistical computing techniques to discover hidden patterns in large data sets by applying statistical methods and algorithms (Han, Kamber & Pei, 2012). Many authors and professionals define data mining as the interdisciplinary interface between the discipline's statistics, software development, and domain knowledge. In fact, Han et al explain, that "the term data mining does not really present all the major components" (Han, Kamber & Pei, 2012) and is just a generalization of tasks that discovers knowledge in large databases. This refers to the synonym knowledge discovery in databases, which includes processes and activities in the fields of data cleaning, data integration, data selection, data transformation, machine learning, classification, clustering, pattern evolution, and knowledge presentation.⁶

One widely-applied reference model, which summarizes the activities of data mining is the CRISP model (Shearer, 2000). It is the continuous and iterative meta-process behind every data mining project. It describes that every data mining task has to build fundamental data understanding and business context, then the data set gets processed with machine learning algorithms before it gets evaluated and deployed. The following figure is the original representation of the CRISP model from Shearer (2000):⁷

⁵ This thesis will be structured based on the five qualitative criteria defined by Bartsch 2015 which are justification, timeliness, originality and valuability (Bartsch, 2015)

⁶ Other authors represent the iterative process of these activities in the famous frameworks CRISP (Shearer, 2000)

⁷ As much in front, text mining is a sub-category of data mining and find its use in the CRISP model only in the data preparation and modeling phase.

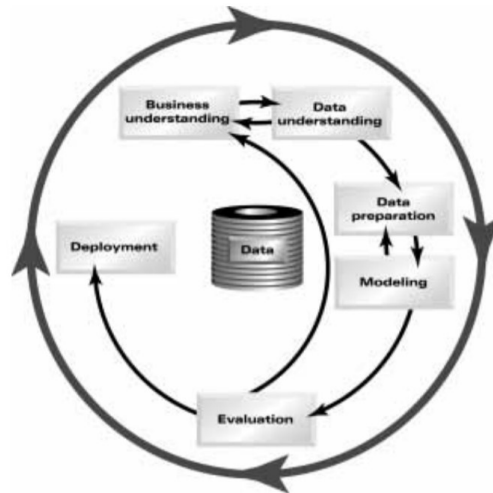


Figure 3 –Graphical representation of the CRISP model⁸

3.1.1. Data structures and data types

The most important component of the CRISP model is the data input. Usually, it has to be extracted from multiple data sources and can be found in three different states, namely structured, semi-structured and unstructured. The term structured refers to organized data, which follows a predefined repository, database structure and defined data types.⁹ Further, the semi-structured data is a structured data set, following specific rules and separators, thus cannot be displayed in relational databases. On the other hand, unstructured data is information, which follows no predefined repository and cannot get stored in a conventional database. Following the knowledge framework of north, in data mining it is a must to scale the unstructured data into granular structured data to perform data mining techniques, as most operational data mining tools, as R Studio, require structured data (Singh, 2018). Another required step to structure a data set is to classify correctly the different data types. This is important as many operations in R or other programming languages require proper data type and data structure declaration to execute functions. In R Studio every data is an object and every object can be declared into six so-called atomic data types, namely: character, numeric, integer, logical and complex. These levels help to distinguish categorical from continuous variables. As the name indicates, categorical variables can label data into different categories. These data declarations are necessary to build informative graphs and charts. On the other hand, continuous variables declare the data as a quantity, which implies, that these data structures, together with discrete variables, are used to perform calculations with the data.

⁸ Copied from Shearer (2000)

⁹ Once declared the variables into factor and continuous variables the data types are defined and structured.

3.1.2. Data mining and Machine Learning

Hence, data mining contains three major techniques, namely i) supervised learning, ii) unsupervised learning, and iii) reinforcement learning. Supervised learning labels data and is mostly performed with clustering and classification techniques. Unsupervised learning is the prediction of unseen data, based on trained machine learning algorithms. Lastly, reinforcement learning is the methodology of gaining real-time models, based on real-time events, which optimizes its parameters based on failure. For this project, only supervised and unsupervised learning techniques will be used and explained during the execution of the stock prediction analysis.

Thereby, the goal of every machine learning algorithm is to pack a set of observations into a universal representative equation, which is known as generalization. This happens by setting the parameters of the target function based on the general concepts from the specific observations, also known as inductive learning (Han, Kamber, & Pei 2012). The goal of inductive learning is to generalize from the training data. The developed equation will be validated in a validation set, and if the predictions are accurate it will be proved in a test set. This ensures the quality and general accuracy of the model on unseen data. In general, if the model is not predicting well it is called underfitting, whereas if it is predicting close to perfect accuracy it is called overfitting. It means that the developed model must find a compromise in between of underfitting and overfitting.

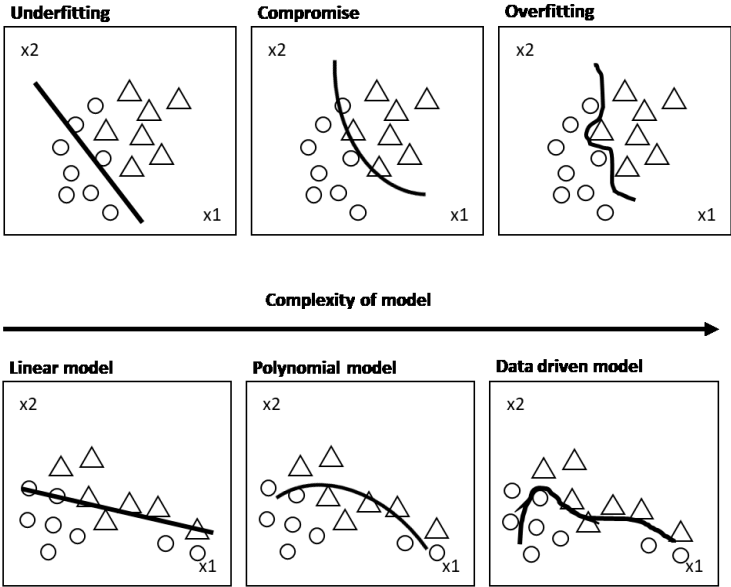


Figure 4 – Phenomenon of underfitting and overfitting

3.2. PREDICTIVE MODELING AND CLASSIFICATION ALGORITHMS

In terms of machine learning algorithms, two methods are identified which will be used in the later project execution, namely a basic approach logistic regression and a complex algorithm neural network. The so-called classification and regression tree (CART) will be also used to conduct other experiments. Both are designed to predict binary variables, as the stock market fluctuation will be calculated with the binary variable named *increase*.

3.2.1. Classification and Regression Trees

The non-parametric classification and regression tree model is a widely applied machine learning algorithm, especially in the text mining context, as it is often used for binary classification problems, with the most applied use case being spam recognition. The CART learns from historic data to build if-then-else splits. By that, it splits the data into recursively smaller subsets, which increases the probability of finding specified classes within the subset. For binominal classification problems, this approach is called recursively binary splitting. Basically, the CART algorithm assigns all training data to the root node and then iteratively it partitions the features by calculating the information gain, or Gini index, or Entropy. The feature with the greatest splitting indicator will be the splitting criterion for the current node. The resulting partition is called a child node. For each child node iteratively it calculates its purity, which checks if the instances are belonging only to one class. If the node is pure, it will declare it as a leaf. If not, it will set the procedure with the current node.

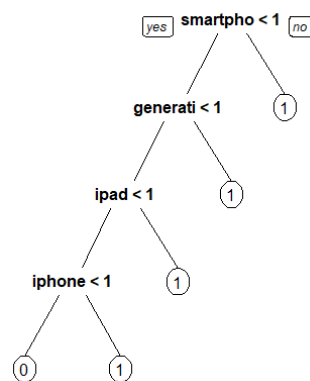


Figure 5 – Illustration of a decision tree

The decision tree approach can be operationalized by using the `rpart` library. The `rpart()` function allows to set up a classification tree effectively. By default, it uses the Gini index to split the different nodes. Also, it provides default parameters to prevent it from growing. In general, decision trees are one of the few machine learning algorithms, which can handle both, continuous and categorical features to predict the dependent variable. In the field of text mining, it can be used to estimate the

importance of the independent variables effectively, particularly to estimate, which tokens are relevant predictors.

3.2.2. Logistic Regression

Logistic regression is another common machine learning algorithm for binary classification problems. Its dependent variable must be categorical, and it is used to find dependencies between independent variables and scores the probability of an unseen observation point being classified to a certain category. Binominal logistic regressions are targeting logical, or also known as binary, classification problems. Basically, binominal logistic regression uses the sigmoid function, which calculates probability scales. Logistic Regression is like a typical linear regression model, with the difference that logistic regression predicts whether data is belonging to the group true or false. Instead of fitting a trend line to the data, logistic regression uses the Sigmoid function to fit a probabilistic curve to the binominal outputs. It means, once the logistic regression model is trained, the probability of unseen data will be used to classify it to the right class.

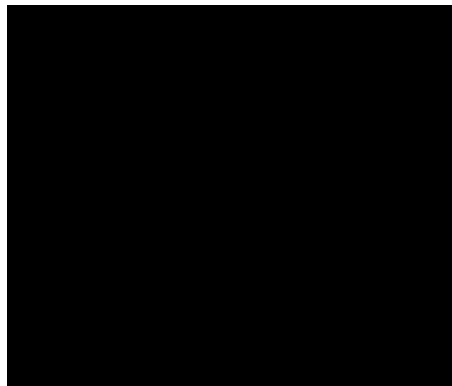


Figure 6 – Illustration of a logistic regression

The benefits of a logistic regression are its simplicity of application, it has low variance, and it provides a probability score for unseen observations. On the counter side, it is only applicable to linear features, which requires high preprocessing time, as categorical variables must be transformed into numeric values. In addition to this, the authors claim that the results of logistic regression moderately drop, when the selected features increase (Agrawal, 2017). For this execution R provides the function `glm()`.

3.2.3. Neural Networks

The data structure of a neural network is an interconnected web of nodes, or also known as neurons, which direct to other neurons (Gaur, n.d.). Neural Networks main function is based on a certain number of features, which compute progressively complex calculations, which address each neuron

with a weight. The weights indicate the importance of the interconnected web, which highlights which neuron must be followed to reach the so-called output layer, which scores the data belonging to a certain class. Gaur explains, that neural networks are commonly used for binary classifications. The idea is to describe the group to which the data observation belongs. Neural Networks are many times referred as deep learning algorithms, because they handle very accurate predictions, especially with a big data set and high number of features. Generally, neural networks are structured in different layers, as the following figure highlights:¹⁰

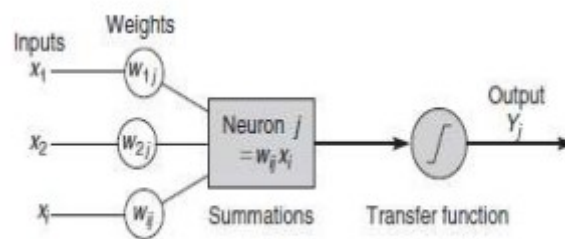


Figure 7 – Representation of Neural Networks¹¹

3.3. FUNDAMENTALS OF TEXT MINING

One example for unstructured data can be found at the present project, namely the variable *text*. Tweets are technically speaking a collection of characters, which cannot be analyzed by machines in its purest form, due to their semantic inability (Han, Kamber, & Pei, 2012). Researching for an applied study to analyze textual information, two research areas could be found, namely natural language processing (NLP) and text mining (Singh, 2018). NLP is the approach to make the machine understand the semantic content behind the textual information, whereas text mining is the preprocessing and translation of the semantic text into numeric values together with analysis (Feinerer, 2008).

3.3.1. Data preprocessing in text mining

The literature describes the purest form of text data as a text corpus, or with other words the raw text. Hereby, the target is to transform the text corpus into numeric values, which can be further processed by the machine. For that purpose, parsing and preprocessing steps are necessary to transform the text into a semi-structured formatted text database (Feinerer 2008). Another aspect of text mining activities is modifying text documents into “categorized, keyword-labeled and time-stamped collection[s]” (Feldmann & Sanger, 2007). Aese (2011) describes important preprocessing

¹⁰ More theoretical context will be given in the Chapter 4.4

¹¹ Copied from Gaur (n.d.)

techniques in text mining, which should be executed in every project, namely preprocessing, cleaning, feature types, feature selection, and reduction. The following figure illustrates Aese phases:

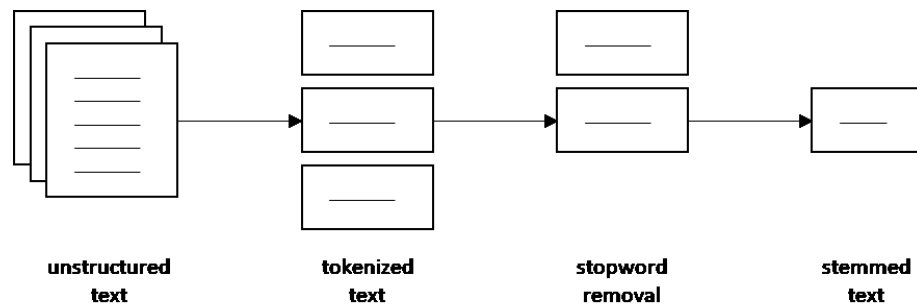


Figure 8 – Stages of text mining stemming techniques

Aese explains that text processing eliminates as much noise as possible to guarantee significant text input. Tokenization is the process of dividing a text into its single elements, e.g. words, phrases, symbols, etc., also known as tokens.¹² After doing so, the previous mentioned stop-word removal deletes insignificant tokens to reduce in a second step the number of features and noise. Last, Aese recommends to perform stemming, which reduces synonyms or acronyms into their semantic root. To name an example the root of the words adjusting an adjustable would be adjusted.

After that, the features of the text have to be extracted to perform data mining techniques. One technique is the n-gram. It is the word-based sequence of n-items from a text input illustrated typically as frequency counts of n-objects, also known as n-gram statistics. Many statisticians found out that 1-grams are due to their simplicity very significant for statistical purposes, but Aese describes that by increasing the n, in particular into a 2-gram, it will contain more information and statistical significance. A technique which complements the n-gram is the noun phrases and proper nouns, which requires a Part-of-speech tagging.¹³ In both cases, a frequency table is built to distinguish important nouns, pronouns, and names grouped by their frequencies to include only highly relevant features. After the preprocessing and the variable modification is executed, feature selection metrics need to be analyzed to guarantee statistical relevant features. For that purpose, Aese proposes three metrics: information gain, mutual information, and chi-square. After granularly building a set of features, the feature reduction process is minimizing the number of features, which will enhance the classifiers for the modeling phase. Many authors describe the well-known singular value decomposition technique, which decreases the data dimensionality of the original feature matrix by representing latent features (Leskovec, Rajaraman, & Ullman, 2016). The following

¹² Many projects tokenized text data into binary variables, to represent for example if a term is included in the observation point. This radically increases features to analyze afterwards the importance for the model

¹³ Part-of-speech tagging labels the string into verbs, adjectives and nouns.

chapters will explain in a brief manner specific text mining algorithms, which will be applied later in the research methodology.

3.3.2. Sentiment analysis

The activities of text categorization and keyword-labeling are also known as sentiment analysis (Mejova, 2009) (Feldmann & Sanger 2007). Sentiment analysis can be defined as a supervised learning technique, which calculates the subjectivity of textual information and transforms it into a quantitative measure or category. In short, the machine understands the polarity between positive and negative emotions by categorizing human language with support of term frequencies and matching systems (Feinerer, 2008). To name an example the well-known polarity sentiment algorithm clusters the text into positive or negative text particles by matching the content with pre-defined or automatically created lexicons. Lexicons represent a list of context-related labeled words. Their purpose is to match the input text with the lexicon list and build a frequency table to indicate the probability for a polarity.¹⁴ The output of this procedure is called term-document matrix (Singh, 2018). Indeed, the lexicon-based approach is one of the most used algorithms for text mining activities. Linoff & Berry described building a lexicon library is the most important and the most time-consuming part of every text mining project (Linoff & Berry, 2011). After searching for such lexica it could be found that R libraries are containing such lists, e.g. stopwords-lists, as in the *tidy* library, which is getting updated consistently by the developers or the R community. However, many authors address that sentiment analysis algorithms are topic-specific and cannot target more complex text data, because of lexicon-based limitations (Mejova, 2009) (Cambira, Das, Bandyopadhyay, & Feraco, 2017). As a complementary algorithm Jones et al 2000 developed an algorithm which distances from the qualitative comparison. The algorithm targets measurements taken directly from the character objects as sentence length, terms frequency, and its discriminativeness from other texts. Hyun et al (2013) summarize these variables within a formula called explanativeness (Hyun Duk Kim, 2013). By that a new dimension can help to translate emotional content into a quantitative indicator. Cambria, Das, Bandyopadhyay & Feraco discuss in their outlook that the field of text mining should conduct research in the field of translating characters into numeric values and combinations to optimize text analytics radically.

To operationalize sentiment analysis, the R community provides the widely applied library *qdap*. It provides a function set, which allows plotting the polarity score of a text variable. Many Authors, especially Singh (2019) claim, that it is one of the most accurate unsupervised NLP technique due to its included context equation. The algorithm of *qdap* assigns a value to a single token of a text value

¹⁴ Also known as bag-of-words mechanism, which identifies the matched words and build depending on that frequency matrixes (Feinerer 2008)

by comparing them with predefined sentiment dictionaries. To not mislead the semantic content of the text variable, a separate context cluster of words is extracted from the text variable. By default it selects the four tokens before and the two tokens after the selected token. With it, the tokens within this context cluster can be tagged as either neutral, amplifier and negator, and weighted using the weights from predefined lexicons. The weights are calculated by summing the product of each polarized and weighted term. A similar and widely applied approach for sentiment analysis is the polarity analysis with the functionality set of the library tidytext. It is a lexicon-based approach with a variety of different lexica, which are scored either quantitative or qualitative. Where Qdap uses equations to score the polarity of a sentence, the tidytext library uses a more simplistic approach. It follows the logic, if the token or text variable contains an element within a lexicon it will be scored as in the lexicon. Three common approaches in tidytext are the AFINN approach, Bing approach, as well as NRC approach. AFINN scores the sentiments with a -5 to 5 ratio. On the other hand, Bing and NRC follow a qualitative approach, by scoring the tokens with predefined values. The research showcased, that both approaches are widely applied in the opinion mining sector. For both scenarios it is important to have a cleaned, preprocessed and ideally tokenized text variable. Another research shows, that qdap is very pruned to large data sets due to its time complexity, whereas tidytext and other libraries have efficient time complexity. Though, to identify opinions and quantify them accurately the qdap library is better, as Singh (2019) consults. The author from qdap provides alternative procedures for conducting accurate sentiment analysis, in particular he recommends the libraries, qdap, syuzhet, sentiment as well as the famous tm package. However, his recommendation for very accurate sentiment analysis is stansent, but this package works only on small data sets.

3.3.3. Latent Dirichlet Allocation

Text mining mostly deals with large data sets, which contains an unmanageable amount of observation points. By that, it is not easy to estimate or understand the semantic context of large text-based data sets. One possibility to extract latent topics within texts or within n-number of documents is the usage of the probabilistic model Latent Dirichlet Allocation. A topic can be represented by a set of words, for example, the words release, date, tomorrow, new, product reminds of the topic "new release". The LDA calculates exactly this approach by randomly allocating tokens from documents into a random topic and afterward estimate their Dirichlet distribution or simplifies their probabilities. The Dirichlet distribution is a multivariate distribution which estimates by random the probability of a mass function. The literature describes the Dirichlet Process, which is the base of this algorithm. It declares k-vector objects and stores randomly a n-text tokens in it. Once n+1 token is placed within the k-vectors, the Dirichlet distribution function will indicate the similarity of the n-tokens. If the Dirichlet distribution of a token contains anomalies, the token will be changed

randomly to another topic. This iteration will continue recursively until the Dirichlet distribution is distributed moderately. In this context two statistical indicators help to control the output of the LDA. The alpha value “alpha controls the mixture of topics for any given document” (Lattier, 2018) and the beta value “controls the distribution of words per topic” (Lattier, 2018). Medium. Lattier (2018) provides a very effective illustration, how the Dirichlet distribution hyperparameter alpha changes, after training the model with 1000 iterations:

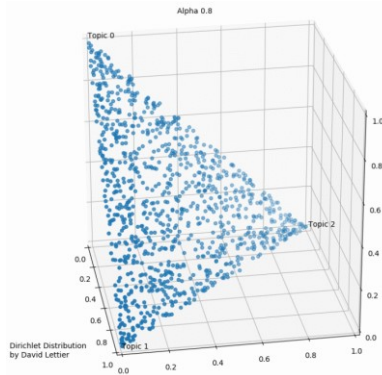


Figure 9 – Dirichlet Distributions after first iteration¹⁵

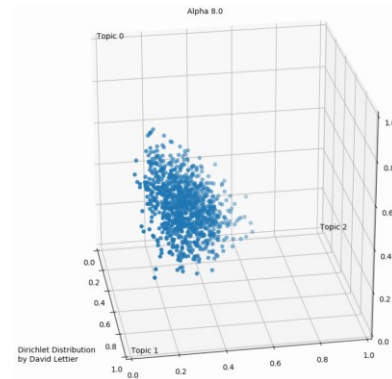


Figure 10 – Dirichlet Distributions after multiple iterations¹⁶

Singh (2019) consults, to increase the k-parameter of the LDA algorithm systematically, to find the right number of topics, which belong within the document. Statistical indicators can be alpha or beta values. Topics can be considered well distributed when there is a high gap between the alpha and beta values. Detailed research found, that there is no best-practice or systematic approach to estimate moderate k-parameter for LDA, other than comparing the alpha or beta values distributions and running experiments by setting up different parameters and iterations.

3.3.4. Limitations with text mining

Searching for existing problems and future works, many participators claim that text mining algorithms aim to overfit the training set, due to its subjective attributes and context-based data. Kao & Poteet (2007) further explain, that “[p]ractitioners of text mining are rarely sure whether an algorithm demonstrated to be effective on one type of data will work on another set of data. Standard test data sets can help compare different algorithms, but they can never tell us whether an algorithm that performs well on them will perform well on a particular user’s dataset” (Kao & Poteet 2007). Also Professor Mahfuz Judeh addresses the context-sensitivity problem by criticizing the lack of scalability of text mining algorithms, as each problem is highly correlated with the domain knowledge input (Judeh, 2018). Judeh recalls that so far no single application has a problem-specific domain knowledge included, e.g. in form of a dynamically adaptable lexicon, and has always to be built separately. In conclusion, this research problem cannot be solved yet. As a matter of fact, there

¹⁵ Copied from Lattier (2018)

¹⁶ Copied from Lattier (2018)

are research projects which addressed this problem with NLP techniques, but the dynamic and not-universal nature of human language will never relief a pattern for data scientist to generalize it into a universal applicable mathematical algorithm.¹⁷

3.4. STOCK MARKET AND PREDICTIVE CAPABILITIES

It is understood that data mining techniques can predict the future based on setting the parameters of an algorithm by collecting historical data and selecting relevant features. Particularly, text mining supports model development and feature selection to build more accurate models.¹⁸ Respecting this knowledge, it can be assumed that a well-trained machine learning algorithm will predict any given target. But the reality shows that many researchers confirm that the stock market is unpredictable, considering the number of experiments for building information systems in this field. In particular the established random walk theory from Cootner (1964) confirms this hypothesis, as Cootner used efficient-market hypothesis to prove it. He also explains that the best strategy to purchase a stock market is to buy and hold the stock for the long-term.

However, a study which consolidated many relevant variables for the stock market price prediction conducted an experiment, where multiple features were combined to enhance the accuracy performance instead of using single features (Tsai & Hisao, 2010). In fact, Tsai & Hisao used in their experiment 85 micro and macroeconomic variables to predict the stock price. The results show that combining multiple feature selection methods, in particular with Principal Components, can provide better prediction performances than using single feature selection methods and reached an accuracy of 73% (Tsai & Hisao, 2010). Although they used many important variables, it can be criticized that no variable was included, which represents the reactivity of the stock price. Addressing the study from Kharb & Malik (2014) the results were that the following qualitative variables are influencing with high importance the stock market and should be considered in every stock market investigation: “[O]verconfidence, herding complex, overreaction, conservatism, preconceived ideas, excessive optimism, representativeness, irrationality or rational way of thinking and the impact of media channels” (Kharb & Malik, 2014). The next chapter will discuss projects which included variables with significant relevance to the dynamic behavior of the stock price, in particular news content, which includes social media contents, like tweets.

¹⁷ This refers to the so-called ambiguity problem. The ambiguity problem describes the capability of understanding a certain information in multiple ways. Considering the multilingual text refinements a universal understanding of a text-corpus is out of the question. ResearchGate (2018) explains this phenomenon in great detail

¹⁸ Obviously, many criteria influence the development of a model, particularly accuracy, robustness, adaption, as well as performance (Giudici & Figini, 2009) (Han, Kamber, & Pei, 2012)

3.5. REFERENCE PROJECTS IN TEXT MINING FOR STOCK PREDICTIONS

While the topic of using social media content to predict the future is widely applied, the literature includes few studies which use advanced text mining methods. A recent study though attempted to use “sentiment indicators created by applying specialized financial microblogging lexicons” (Oliveira, Cortez, & Areal, 2015). Oliveira et al explain that most of the existing studies used a relatively smaller data set with a few sets of variables. Oliveira et al studies included a variety of correlated twitter variables as well, namely a daily twitter indicator in addition to other news contents like “weekly American Association of Individual Investors and Investors Intelligence (II) values and monthly University of Michigan Surveys of Consumers and Sentix values” (Oliveira, Cortez, & Areal, 2015). They explain in their outlook that twitter data are analyzed more in the advertisement field then rather focusing on translating the content into features for stock market predictions.¹⁹ To optimize future predictions, they intend to also include domain knowledge from other social media channels to train the model with relevant variables from different sources.

Nevertheless, the procedure of Oliveira et al is more an exception than a rule. The common denominator in the field of text mining prediction is sentiment analysis as it is detected that most of the studies used different types of sentiment analysis. Sentiment analysis, also known as opinion mining, represents a complex activity in the area of quantifying people’s digital behavior, but mainly analyses emotions, opinions, attitudes, etc. (Liu, 2012). It is subdivided into the field of NLP – the study which processes the human language to computers. An important insight by Liu 2012, Oliveira, Cortez, & Areal 2015 and Jadhav & Wakode 2017 is very prone to spam, as well as outliers like unrelated content.

Addressing again the unpredictability of the stock market, a study was found which confirms that it is not possible to predict future fluctuations up to 50% accuracy, but “very early indicators can be extracted from online social media [...] to predict changes in various economic and commercial indicators” (Bollen, Mao, & Zeng 2010). In their research, this was achieved by filtering only relevant tweets with the premise that the tweet content is strongly correlated with the stock market. This was achieved by using regular expressions, polarity sentiments, Google-Profile of Mood States, as well as aggregated public mood sentiments and filters. Bollen et al determined that by increasing the dimensions of sentiment analysis the predictive power of the classifiers will increase and recommend this method for future projects.

¹⁹ Also other studies indicate that twitter data are mainly used for advertisement analytics.

3.6. APPLIED R LIBRARIES

As the present project will be executed with the programming language R, it was necessary to find reference projects, which applies the previously mentioned data mining techniques. Particularly, the Cambridge University data scientist Minerva Singh (2018) summarizes different text preprocessing and modeling techniques on twitter data, which will help to construct efficient R code and understand the data structures in R, as well as understand the functionalities of diverse text mining libraries in R studio. Respecting all the different approaches described in the state of arts chapter, the following text summarizes a selection of the most important libraries, which will be used in to perform text and data mining activities. In total over 80 libraries and packages were used.

Table 1 – Selection of applied R libraries

Library	Author	Description
library("plyr")	(Wickham, 2018)	This package is used to apply merge functions
library("tm")	(Feinerer, 2008)	This package is used to apply text mining activities and to build text mining relevant data types as term-document matrices
library("SnowballC")	(Bouchet-Valat, 2018)	This package is used to perform text mining modifications, as Porters stemming algorithm
library("ggplot2")	(Wickham, et al., 2018)	This package is one of the most established packages for data visualization and graphics. It is used to perform all the visualizations of this project
library("RSentiment")	(Subhasree & Groswami, 2018)	This package is used to perform NLP functions as sentiment analysis
library("readr")	(Wickham, Hester, Francois, & Jorgensen, 2018)	This package is used to import and export csv data
library("wordcloud")	(Fellows, 2018)	This package is used to visualize the content of the twitter data, to build word clouds and illustrate the frequencies
library("qdap")	(Goodrich, Kurkiewicz, & Rinker, 2018)	This package is used to perform qualitative and quantitative analysis, frequency counts and term counts
library("tidytext")	(Silge & Robinson, 2018)	This package contains important functions for text mining activities as stop word lists, n-grams, etc.
library("RColorBrewer")	(Neuwirth, 2018)	This package is a complementary function for tailor-coloring graphs and figures
library("dendextend")	(Galili, 2018)	This package offers functionalities to create and to enhance n-grams and dendrograms
library("topicmodels")	(Grün, 2018)	This package is used to perform Latent Dirichlet Allocation models and Correlated Topics models
library("dplyr")	(Wickham, Francois, Henry, & Müller, 2018)	This package is used to perform mutation functions, as rearranging the data frame structure
library("pastecs")	(Grosjean, Ibanez, & Etienne, 2018)	This package is used to transform irregular time series into regular time series
library("quanteda")	(Beniot, et al., 2018)	This package provides a function set for the quantitative analysis of textual content.
library("neuralnet")	(Fritsch et al., 2019)	This package provides a function set to train, modify, perform and analyze neural networks.

4. PROJECT METHODOLOGY AND EXECUTION

The following chapter will discuss the research area of data mining and showcase important techniques and methodologies, which will be conceptualized later in this chapter. Specifically, text mining techniques will be explained, and statistical measurements will be presented, which will complement the literature review. To secure a systematical approach, the following chapters will include the executed R script. The following figure represents the technical methodology and summarizes all procedures which are described in the next chapters.

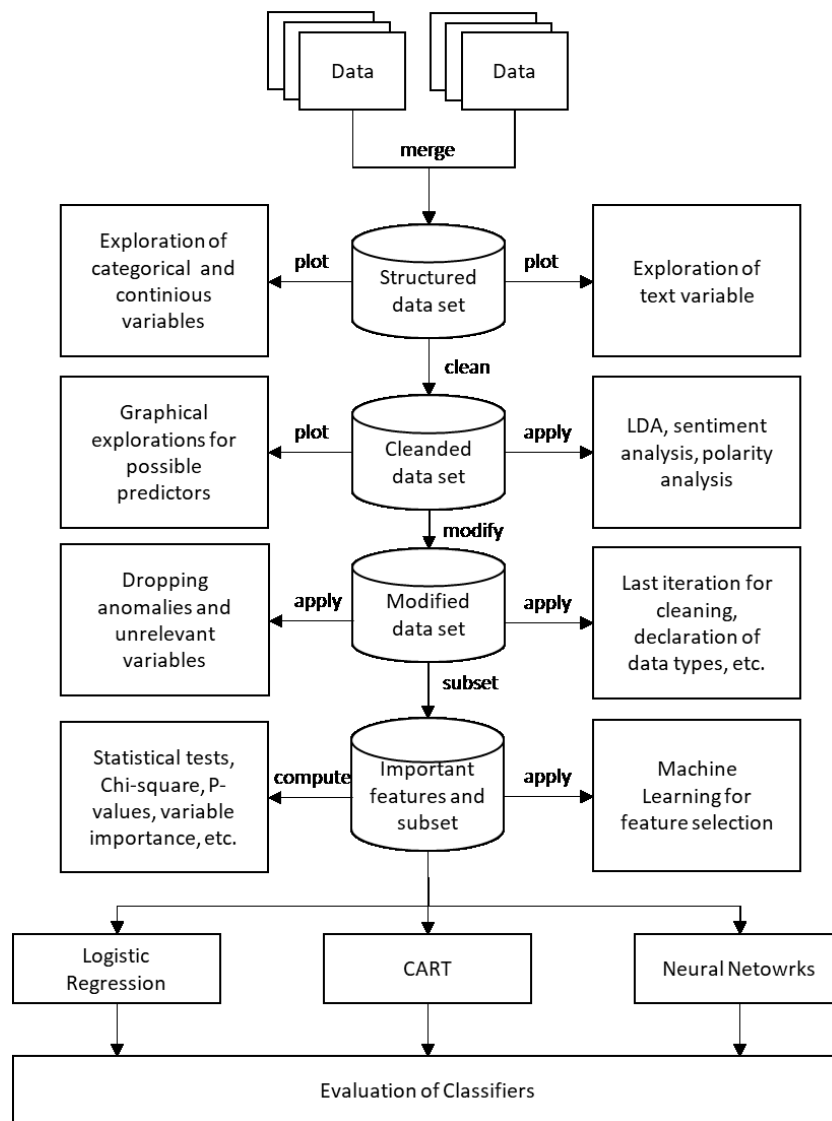


Figure 11 – Methodology and structure of present project work

4.1. DATA PREPROCESSING

This chapter addresses the project scope “Preprocess the twitter data and perform data cleaning tasks to reduce noise” by extracting the delivered csv files and merge them with the extracted yahoo finance data. Data issues will be examined to discover anomalies and to solve them with data modification scripts. All data types will be viewed and declared to allow a smooth transition to the statistical exploration phase. The overall goal is to build a cleaned data set, which will be the foundation for further project execution.

4.1.1. Extract and Preprocess the Data into R Studio

The present twitter data from the companies Amazon, Apple, Facebook, Google, Microsoft, and Oracle were sent separately as multiple comma-separated-value (CSV) by NOVA IMS, which simplified the data repository in three variables, namely: Identification/ID, unstructured tweet content, as well as an unstructured timestamp. On first glance, it contains a different variable structure as the original twitter API provides, which means there is a loss of important information for this project “Twitter Developer Docs” (n.d., 2019). It is assumed, that the twitter data set was preprocessed beforehand, without finalizing important tasks as cleaning the text and applying modifications to the timestamp. Furthermore, comparing the Twitter API “Twitter Developer Docs” (n.d., 2019) (Singh, 2018), many variables were dropped beforehand, which increases the information loss even more, in particular variables as publisher, handle, is_retweet, language, etc.²⁰ However, to consolidate over 100.000 CSV files the Windows command tool was used to merge the multiple files into one CSV file, which contained over 300 Megabytes of data and in total over 1.7 million observation points. With it, it is possible to examine and to preprocess the data into R Studio.

4.1.2. Explore Data Issues and apply Data Cleaning

In first glance, the timestamp, as well as the text variable, are unstructured. For that sake, they must be transformed to ensure the quality of the data preprocessing. The first activity is to split the timestamp in such a manner, to use Twitter timestamp as a foreign key to merge them with the downloaded stock data from the Yahoo API. The following R code operationalizes this procedure:

²⁰ This observation was taken from a extracted Twitter data set from Singh (2018)

```
#Created_at must be restructured and split into different factors.
#Drop the factor variable random as it provides no information.
#Tidyr provides separation function.
library(tidyr)
train <- separate(train, 'created_at',
  into = c("dayname", "month", "day", "time", "random", "year"),
  sep = " ", remove = TRUE, convert = FALSE)
train$length_tweet <- nchar(train$text, type = "chars", allowNA = FALSE, keepNA = NA)
train$Month <- match(train$Month, month.abb)
train$Random <- NULL
```

With this code, it is possible to restructure the variable `created_at` into 6 new variables. Simultaneously, it allows using `date` as a foreign key to merge the data set with the extracted data from the Yahoo API. The following data model describes the AS-IS state:

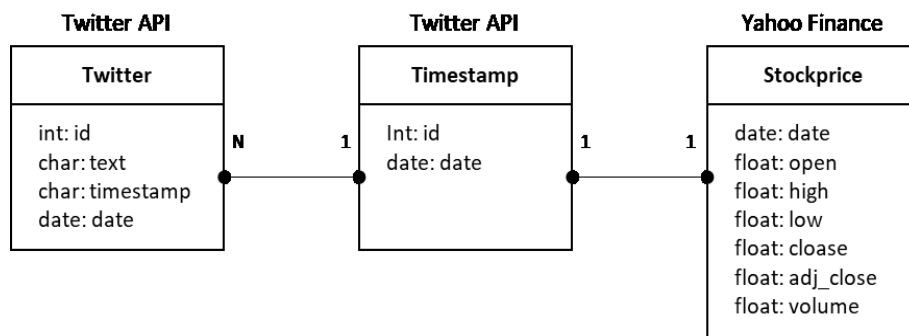


Figure 12 –Data model for the present project

With the setup of the entity-relationship model, one problem becomes clear. The present database has multiple tweets for each day, but on the other hand for each day only one stock price information exists. For that sake, the twitter data for each day will be merged with one stock price, and the binary variable *increase* will be added to the stock price data set, which indicates if the stock market raised or decreased in comparison to the previous day. Simultaneously the variable *increase* is declared as the target variable.²¹

The biggest issue, which is observed in each company’s tweets, is the page break within the variable *text*, which necessarily involves identifying and dropping broke observation points from the data set in the ETL process, as neither R Studio nor Excel provides functions to eliminate page breaks within the loading phase. Many authors state to not delete outliers without operating replacement methods, but it is decided to drop the missing values as they contain roughly 1% of the data (Viljamaa, 2017). Authors claim that it is necessary to simplify semantic text to build in a later stage a

²¹ Later this will be discussed in detail and further modifications will be applied (compare Chapter 4.2).

corpus for the sentiment analysis. For that reason, the library *gsub* will support the process to clean the text values and simplify them for each company, respecting the requirements of Bollen, Mao, and Zeng (2010) as well as Mejova (2009). They all discuss that the text must be screened for noise, which will have negative impact on the computation power. By cleaning insignificant terms and symbols the algorithms can detect qualitatively semantic contents. This is the reason, why it is crucial to clean the text from hyperlinks, digits, profile tags, punctuations, hashtags, as well as numbers. Other steps were to lower case the text, to get rid of white spaces, and to exclude stop words. In text mining, stop words are the most common words in any given language, which contain no semantic signal for the applied algorithms. The following script applies this method:

Script 2 – Deleting NA and cleaning the text variable

```
#Cleaning the text variable based on Singhs Project
train$text <- tolower(train$text)
train$text <- gsub("http[^\s:]+", "", train$text)
train$text <- gsub("[[:digit:]]", "", train$text)
train$text <- gsub("[[:punct:]]", " ", train$text)
train$text <- gsub("#.*", "", train$text)
train$text <- gsub("@.*", "", train$text)
train$text <- gsub("[\r\n]", "", train$text)
colnames(train) <- tolower(colnames(train))
train$text <- gsub("[\r\n]", "", train$text)
```

Another problem is the assembled and unstructured *timestamp*. It is necessary to split it, as it represents the foreign key for the merge, as well as to increase the variable input for further statistical explorations. As a result, the timestamp is split into *dayname*, *month*, *day*, *time*, *year*, and the aggregated and structured *date* variable. With it, the amount of factor variables increases for analysis purposes, and the twitter data set is ready for the merge with the queried and downloaded yahoo finance API data set for each company. The following R script merges the twitter data set with the yahoo API data set.

Script 3 – Merging the twitter and yahoo data set

```
#align headers and align format to prepare for merge.
#merge the stock with the tweet data.
summary(train$date)
colnames(facebook_stock) <- tolower(colnames(facebook_stock))
str(facebook_stock$date)
facebook_stock$date <- as.character(facebook_stock$date)
facebook_stock$newdate <- strptime(as.character(facebook_stock$date), "%d.%m.%Y")
facebook_stock$newdate <- format(facebook_stock$newdate, "%Y-%m-%d")
facebook_stock$newdate <- as.Date(facebook_stock$newdate)
facebook_stock$date <- facebook_stock$newdate
facebook_stock$newdate <- NULL
facebook_tweet_stock <- merge(train, facebook_stock, "date")
```

The twitter data sets are extracted in the time of 29th November 2016 till the 6th December 2016. This time is selected for every company at the yahoo finance API website and with it, the following

variables will be: *date, open, high, low, close, adj_close*, as well as *volume*. An important fact is that Yahoo excludes stock data from two days, namely the third and fourth of December. Additionally, the variable *increase* will be added to the data set to identify, if the stock increases in comparison to the previous day. After applying these actions and formatting the variables, both data sets are merged by using the *date* variable as the foreign key. Before finalizing this procedure, it is necessary to standardize the numeric values by removing unnecessary currency stamps and symbols. Alongside, the data types for continuous and categorical variables will be declared.

4.1.3. Preprocessed Data Set

In summary, over 200 thousand observations and missing values must be eliminated, which represents ~10% of the data set. Respecting many authors investigations, this amount of data points can have a great impact on any data mining project, but due to the hidden page breaks it is not possible to find another workaround, neither in the Windows Console, neither in Excel nor in R Studio. The following data set builds the foundation for further analysis. It contains 12 variables and a total of 1.592.607 observation points.²²

Table 2 – Variables of first data set

Variable name	Variable type	Description
Id	Numeric	Identification of observation point
date	Date	Record date of twitter post and stock market
text	Text	Twitter content
dayname	Factor	The name of the day of the belonging date
month	Factor	Month name of the belonging date
day	Factor	Day number of the belonging date
time	Timestamp	Time of twitter post
length_tweet	Integer	Length of twitter content
company	Factor	Name of the belonging company
open	Numeric	The opening price of stock market on belonging date
high	Numeric	The highest value of the stock market on belonging date
low	Numeric	The lowest price of stock market on belonging date
close	Numeric	The closing price of stock market on belonging date
adj_close	Numeric	The adjusted closing price of the stock market on belonging date
volume	Numeric	The volume of the stock market on belonging date
Increase	Logical/ Target Variable	Recorded as “1” if the stock market increased in comparison to the previous date

4.2. DATA EXPLORATION

This phase is an important step for the further procedure, as it builds essential domain knowledge and targets a great amount of the project scope, namely: “Preprocess the twitter data and perform

²² The code of the above mentioned actions can be found in the annex

data cleaning tasks to reduce noise” and “Apply text mining techniques to quantify the unstructured text into relevant features”. Statistical and graphical explorations will be addressed separately for continuous, categorical and text variables. The target is to find evidence for discriminators, correlations, predictors, as well as to describe the content of the text variable. The resulting insights will be used to build relevant features and to apply important modifications as well as statistical experiments.

4.2.1. Examination of continuous variables

The data set contains two sort of variables, namely 11 continuous variables, particularly numeric and integers, as well as 6 categorical variables, particularly logical and factors. To showcase their distributions and to identify anomalies the following univariate analysis concentrates on the continuous variables. The following table summarizes the central tendencies:

Table 3 - Univariate analysis of the first data set

Variable	Min.	1 st Q	Median	Mean	3 rd Q	Max	St. Dev.
month	11.0	11.0	12.0	11.7	12.0	12.0	0.46
year	2016.0	2016.0	2016.0	2016.0	2016.0	2016.0	0
length_tweet	5.0	56.0	101.0	97.1	126.0	9881.0	41.65
open	38.46	111.60	118.38	354.24	752.41	771.53	312.99
high	38.83	112.20	118.45	357.10	759.85	778.50	315.558
low	37.64	110.27	116.33	350.12	742.00	768.24	309.56
close	38.5	111.5	117.4	353.2	750.5	770.8	312.27
adj. close	37.29	108.07	117.43	352.34	750.50	770.84	312.94

The central tendencies, especially the standard deviations, show that most of the variables are skewed. This is related due to the 1:n (one-to-many) relationship of the twitter data set and the stock data set. Also every company contains different stock prices, which indicates that it is distant from the mean.²³

Furthermore, the correlations between the continuous variables were examined to understand if the daily stock market prices will have any positive or negative correlations with other features. Pearson’s correlation is the correlation method, which evaluates continuous variables (Han, Kamber, & Pei, 2012). The following R code calculates the Pearson Correlation plot:

²³ Usually this implies to use normalization or standardization functions, which will scale the data into a given range to make it comparable (Han, Kamber, & Pei, 2012). It is important to mention, that in a later stage the variables *open*, *high*, *low*, *close* and *adj.close* will be dropped for the further analysis.

```
library(corrplot)
correlations_stock <- modify[,c("id_str", "month", "day", "length_tweet",
                              "open", "high", "low", "close", "adj.close", "company_number")]
correlations_stock$increase <- as.integer(as.factor(correlations_stock$increase))
pearson <- cor(correlations_stock, method = c("pearson"))
corrplot(pearson, method="number", tl.col = "black")
```

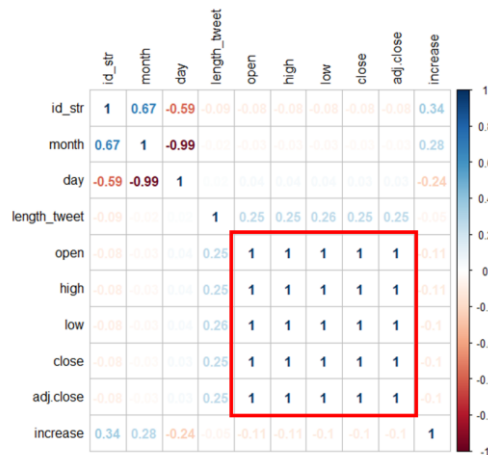


Figure 13 – Pearson correlation before the data modification

The figure showcases that the data set has no significant correlations yet. As assumed, the stock market variables have a coefficient of 1, which make them insignificant for further analysis. As a result, the stock market data will be dropped and only the target variable *increase* will remain.

4.2.2. Examination of Outliers

The high standard deviation of the stock data is related due to the different stock market prices each company has. Another important observation is the skewed standard deviation of the variable *length_tweet*. Comparing the mean, the maximum and the standard deviation value of the central tendencies, it crystalizes that the tweet contents contain outliers. The following boxplot highlights the distribution of the variable *length_tweet*. The plots were built by using the library *ggplot2*, especially by following the next code architecture:

```
ggplot(data = modify_final,
       aes(x=modify_final$company,
          y=modify_final$length_tweet)) +
  geom_boxplot(aes(fill=modify_final$increase)) +
  xlab(label = "Company") +
  ylab(label = "Tweet length") +
  theme(legend.title=element_blank())+
  ylim(0,200)
```

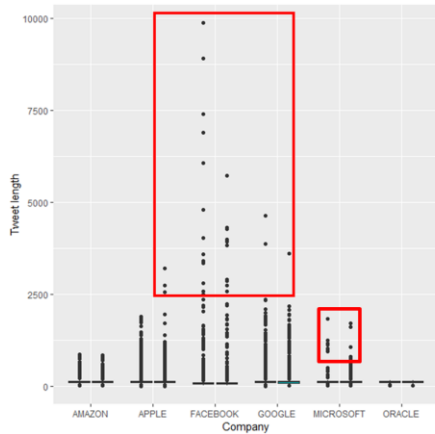


Figure 14 – Boxplots indicate outliers

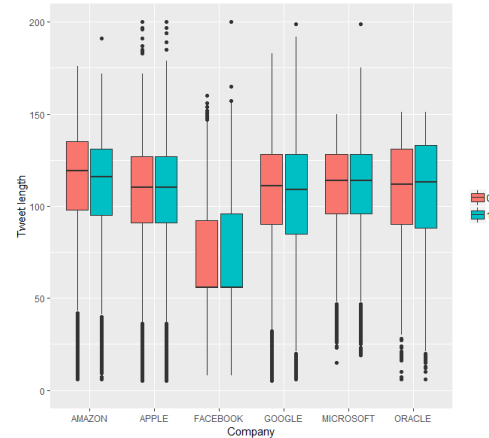


Figure 15 – Boxplots for length of tweets

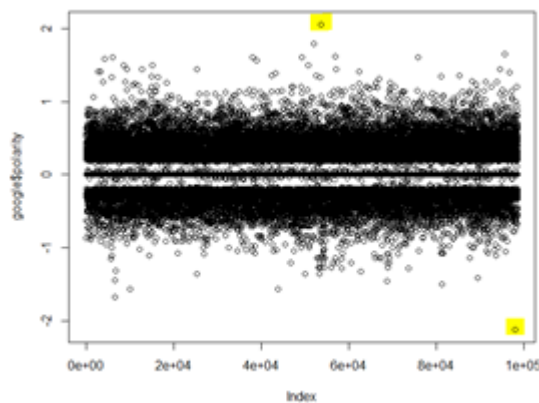


Figure 16 – Outliers with heavy content

Figure 14 highlights clearly the outliers of the variable *text*. A threshold of 3.000 was defined to describe the content of these tweets by subsetting this threshold as a new data frame. The top 25 most frequent terms of this data frame show tendencies for bot messages, as well as automated Facebook updates, as the following word cloud showcases (Figure 17):²⁴

Script 6 – Building word clouds to examine the content of the outliers

```
library(wordcloud)
library(tm)

threshold_outliers <- subset(final, final$length_tweet > 3000)
word_corpus_outliers <- Corpus(VectorSource(threshold_outliers$text))
word_corpus_outliers <- tm_map(word_corpus_outliers, stripwhitespace)
word_corpus_outliers <- tm_map(word_corpus_outliers, removewords, stopwords('english'))
wordcloud(word_corpus_outliers,
          scale=c(5,0.5),
          max.words=25,
          random.order=FALSE,
          rot.per=0.35,
          use.r.layout=TRUE)
```

²⁴ This observation will be targeted in great detail in the Chapter “Exploration of the text variable”



Now the distribution for each company will be highlighted. Visualizing its count and distribution with different plots it becomes clear, that so far none of the existing variables show moderate discriminative signal. All the following figures were built by using the library ggplot2, particularly by taking the following example as a reference:

Script 7 – Exploring the target variable

```
#examination of the variable increase within different companies
ggplot(modify_final, aes(x= as.factor(modify_final$company),
                          y=frequency(modify_final$ID)))
+ facet_grid(~modify_final$increase)
+ xlab("Company") + ylab("Frequency")
```

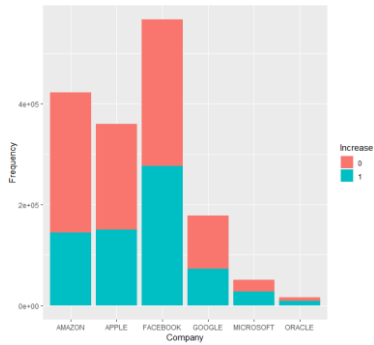


Figure 19 – Distribution of variable Increase based on company

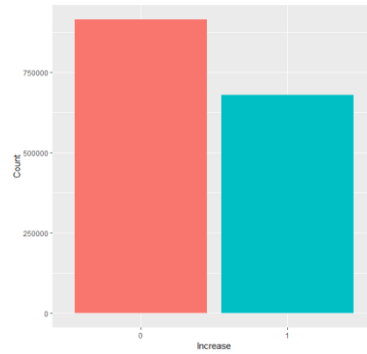


Figure 20 – Count of variable increase

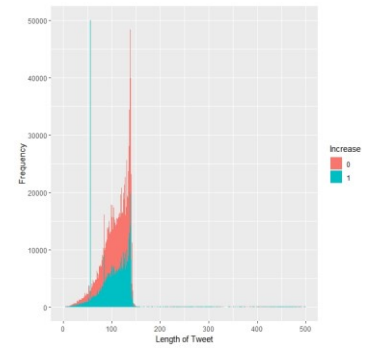


Figure 21 – Distribution of variable Increase based on length of the tweets

The *length_tweet* frequencies show when the stock market increases the *length_tweet* slightly decreases (compare Figure 20).²⁵ To further investigate this phenomenon a drill-down is necessary to investigate this behavior for each company. For that sake, a bar chart will show, how the tweet length will distribute for increased and decreased stock prices for each company. With the following figures it can be observed marginally, that the outlier peaks increase once the stock market decreases.

Script 8 – Investigating the distribution of the tweet lengths

```
#Tweet_length distribution for oracle
ggplot(final_oracle, aes(x=final_oracle$length_tweet,
                        y=frequency(final_oracle),
                        fill=as.factor(final_oracle$increase))) +
  geom_bar(width = 1, stat = "identity") +
  xlim(0,200) + ylim(0,200) + facet_grid(~final_oracle$date) +
  xlab("Length of Tweet") + ylab("Frequency") +
  guides(fill=guide_legend(title="Increase"))
```

²⁵ Respecting the previous findings, it can be hypothesized, that when the stock price increases less bot messages will appear. This hypothesis will be discussed in the chapter “Exploration of the text variable”

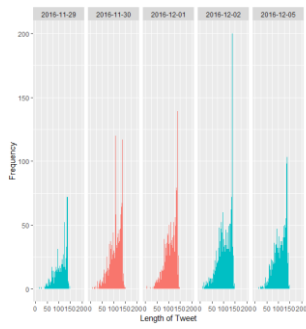


Figure 22 – Distribution of Increase through date for Oracle

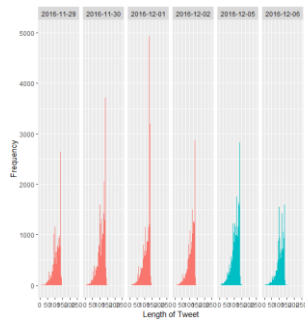


Figure 23 - Distribution of Increase through date for Amazon

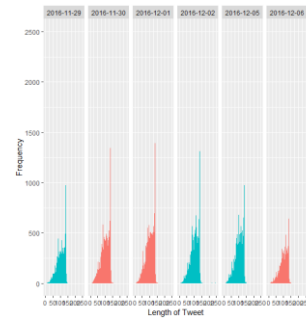


Figure 24 - Distribution of Increase through date for Facebook

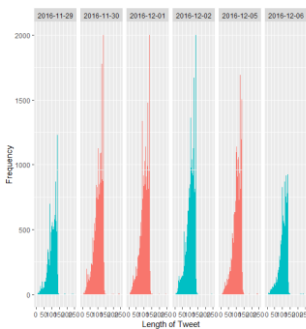


Figure 25 - Distribution of Increase through date for Apple

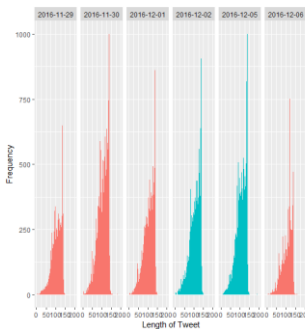


Figure 26 - Distribution of Increase through date for Google

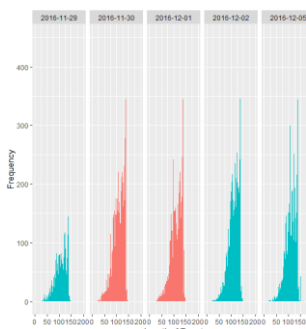


Figure 27 - Distribution of Increase through date for Microsoft

To further analyze this behavior, another drill-down will be executed. For the first time in this report the *time* variable will be targeted by extracting the *hour*. The following code, and with it the resulting figure, illustrates the outcome:

Script9 – Performing drill-downs to examine the target variable

```
# Drill down for each company
final_microsoft$hour <- substr(final_microsoft$time, 0, 2)
final_microsoft$hour <- as.integer(as.character(final_microsoft$hour))
ggplot(final_microsoft, aes(x=final_microsoft$hour,
                           y=frequency(final_microsoft,
                                       fill=as.factor(final_microsoft$increase)))) +
  geom_bar(width = 1, stat = "identity") +
  ylim(0,1000) +
  xlim(0,23) +
  facet_wrap(~final_microsoft$date) +
  xlab("Hour of the day") +
  ylab("Frequency")
```

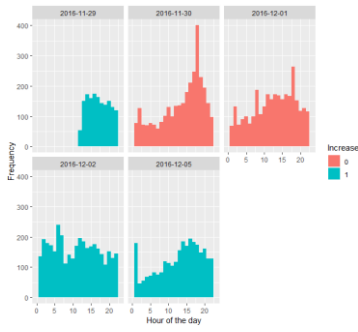


Figure 28 – Oracle tweets frequencies for each hour per day



Figure 29 – Amazon tweets frequencies for each hour per day



Figure 30 – Facebook tweets frequencies for each hour per day

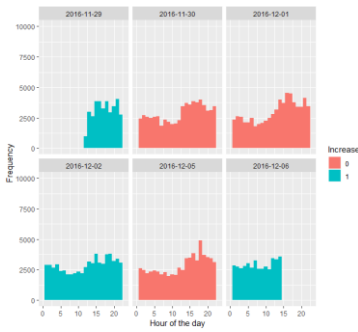


Figure 31 – Apple tweets frequencies for each hour per day

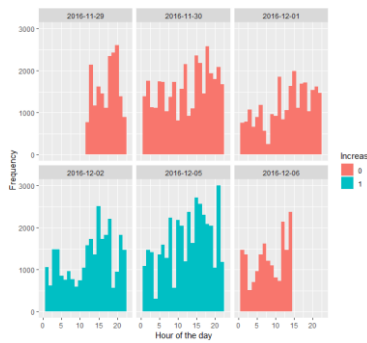


Figure 32 – Google tweets frequencies for each hour per day

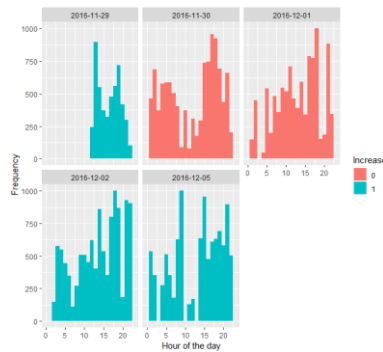


Figure 33 – Microsoft tweets frequencies for each hour per day

The drill-down shows the previous phenomenon from a different angle. It appears that the distribution scatters evenly throughout the day and reaches its peak around 5 pm for each company. Indeed, the roll-up shows a moderate increase of tweet streams, when the stock market drops, with exception of 2016-11-29 and 2016-11-30. As it is difficult to derive a clear assumption, other plots helped to score the number of activities by aggregating the data. The graph 34 shows that the frequency increases indeed when the stock market drops. On the other hand, a drill-down contradicts this hypothesis. The last figures will be targeted in a later stage, once more factors will be built.

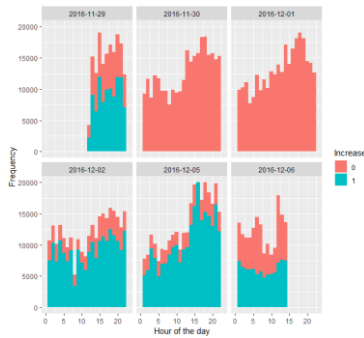


Figure 34 – Tweet frequencies for each hour per day

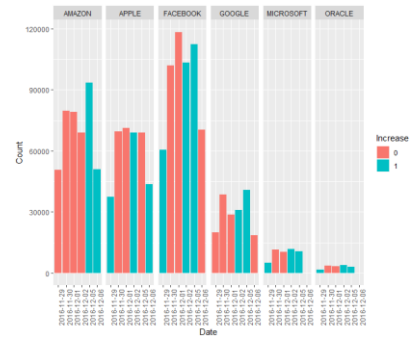


Figure 35 – Count of increased values by trading hours

4.2.4. Exploration of the text variable

To visualize the content of the variable *text*, respecting that the head of the data frame, and the categorical and outlier analysis, show evidence for bots, spams, two separate data sets will be used, namely, a data set with unique tweets and one with duplicate tweets. The following word cloud and frequency table inferences, that the *text* variable indeed includes bot terms, which most are belonging to the companies Amazon, Apple, Facebook, and Google.

Script 10 – Building word clouds and frequency tables to examine word frequencies

```
#wordcloud and word frequency table
library(wordcloud)

word_corpus <- Corpus(VectorSource(outliers$text))
word_corpus <- tm_map(word_corpus, stripwhitespace)
word_corpus <- tm_map(word_corpus, removeWords, stopwords('english'))

tdm <- TermDocumentMatrix(word_corpus, control = list(minwordLength=c(1,Inf)))
tdm_subset <- removeSparseTerms(tdm, sparse = 0.90)
m <- as.matrix(tdm_subset)

# Plot frequent words
freq <- rowSums(m)
freq <- subset(freq, freq>=200)

wordcloud(words = final$text,
           freq = x$n, max.words = 150,
           random.order = FALSE,
           colors = brewer.pal(8, "Dark2"))
```


them into k-groups of topics. As the LDA algorithm is context-dependent, it will be performed for each company. The LDA k parameter will be declared as 5, with 500 iterations, and a random seed, as the following script showcases:

Script11 – Performing LDA to output the top 10 topics

```

library(quantda)
duplicates_facebook <- duplicates_final[which(duplicates_final$company == "FACEBOOK"),]
duplicates_amazon <- duplicates_final[which(duplicates_final$company == "AMAZON"),]
duplicates_google <- duplicates_final[which(duplicates_final$company == "GOOGLE"),]
duplicates_apple <- duplicates_final[which(duplicates_final$company == "APPLE"),]
duplicates_microsoft <- duplicates_final[which(duplicates_final$company == "MICROSOFT"),]
duplicates_oracle <- duplicates_final[which(duplicates_final$company == "ORACLE"),]
duplicates_final <- subset(modify_final, modify_final$duplicate==1)
head(duplicates_final)

#script for LDA
corpus_duplicates <- corpus(as.character(duplicates_oracle$text))
dfm_duplicates <- dfm(corpus_duplicates, ngrams=1L, remove = c(company_stopwords, stopwords("english")))
vdfm <- dfm_trim(dfm_duplicates, min_termfreq = 10, min_docfreq = 5)
topfeatures(vdfm, n = 10)
dtm <- convert(vdfm, to = "topicmodels")
K <- 5
lda <- LDA(dtm, k = K, method = "Gibbs", control = list(verbose = 25L, seed = 123, burnin = 100, iter = 500))
term <- terms(lda, 5)
term

```

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"support"	"amp"	"job"	"cloud"	"corporation"
[2,]	"new"	"class"	"ca"	"via"	"business"
[3,]	"aws"	"media"	"developer"	"finance"	"dyn"
[4,]	"goes"	"digital"	"database"	"netsuite"	"tech"
[5,]	"postgressql"	"billion"	"opportunity"	"global"	"resources"

Figure 40 – LDA and top features Results for Oracle

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"xbox"	"new"	"surface"	"xbox"	"windows"
[2,]	"one"	"office"	"gt"	"console"	"home"
[3,]	"amp"	"cloud"	"studio"	"gb"	"echo"
[4,]	"game"	"android"	"pc"	"black"	"via"
[5,]	"support"	"k"	"mobile"	"games"	"microsofts"

Figure 41 – LDA and top features results for Microsoft

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"ipad"	"iphone"	"watch"	"now"	"gb"
[2,]	"wi"	"smartphone"	"win"	"store"	"black"
[3,]	"fi"	"unlocked"	"via"	"music"	"generation"
[4,]	"space"	"gold"	"says"	"apps"	"ipod"
[5,]	"gray"	"plus"	"cook"	"new"	"th"

Figure 42 – LDA and top features results for Apple

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"play"	"giveaway"	"android"	"home"	"now"
[2,]	"app"	"pixel"	"via"	"search"	"apps"
[3,]	"store"	"international"	"gt"	"nexus"	"playing"
[4,]	"darksummoner"	"x1"	"accounts"	"maps"	"seo"
[5,]	"today"	"androidauth"	"earth"	"echo"	"radio"

Figure 43 – LDA and top features results for Google

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"gift"	"amp"	"just"	"ad"	"win"
[2,]	"card"	"new"	"check"	"amazongiveaway"	"via"
[3,]	"win"	"buy"	"via"	"enter"	"fire"
[4,]	"giveaway"	"echo"	"participated"	"free"	"kindle"
[5,]	"enter"	"price"	"go"	"sale"	"black"

Figure 44 – LDA and top features results for Amazon

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
[1,]	"new"	"photos"	"posted"	"new"	"photo"
[2,]	"posted"	"like"	"new"	"posted"	"posted"
[3,]	"photo"	"page"	"video"	"photo"	"new"
[4,]	"check"	"live"	"photo"	"friends"	"intelligence"
[5,]	"twitter"	"album"	"fake"	"three"	"artificial"

Figure 45 – LDA and top features results for Facebook

It appears that the parameter settings cluster the top 5 topics of each segment confidently, except on Facebook. The Facebook subset has no clear semantic boundaries, as all the 5 segments include wording from the same word root. No changes appear, when synchronizing the parameter setting by increasing the iterations, due to the fact, that almost all duplicates, in particular over ~95%, refers to the tweet “I posted a new photo to facebook”. Despite, all the other results show terms and group of words, which relates to advertisements as hypothesized. Before resulting in a specific action the unique values will undergo the same experiment.

4.2.4.2. Topic exploration for unique text values

For the unique values, the most frequent terms function and LDA will be performed. Actually, the LDA topics were not distant from the duplicate topics, as again the majority include advertisements for products. The following summary presents the most frequent words for each company:

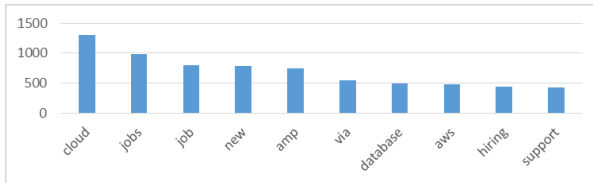


Figure 46 – Top 10 words of Oracle

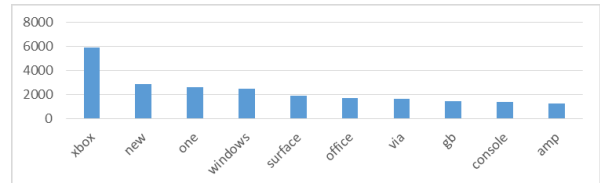


Figure 47 – Top 10 words of Microsoft

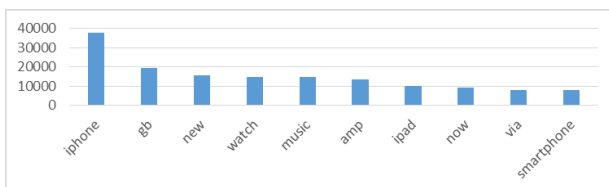


Figure 48 – Top 10 words of Apple

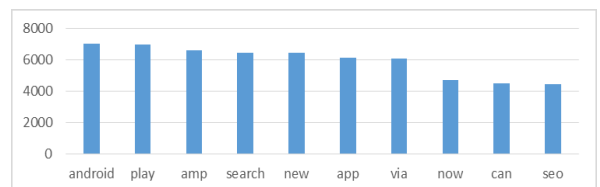


Figure 49 – Top 10 words of Google

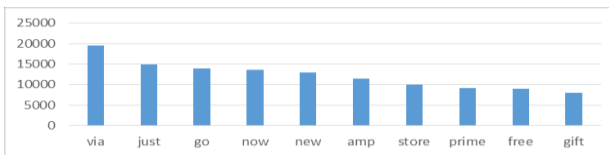


Figure 50 – Top 10 words of Amazon

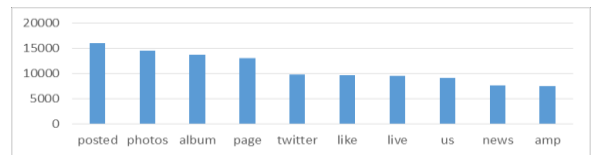


Figure 51 – Top 10 words of Facebook

Although the plots represent unique values, they contain terms, which skew the data, for example, the photo update from Facebook via Twitter and Apples product iPhone. To further analyze the most frequent terms, the produced LDA matrix will be used to examine the word-topic probabilities by transforming the LDA matrix into a data frame and afterward applying a grouped ggplot, as the following code showcases:

Script 12 – Building based on the LDA result the word-probability plots

```
### word topic propabilities
library(tidytext)
unique_topics <- tidy(lda, matrix = "beta")
unique_topics
library(ggplot2)
library(dplyr)
unique_topics_terms <- unique_topics %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
unique_topics_terms %>%
  mutate(term = reorder(term, beta)) %>%
  ggplot(aes(term, beta, fill = factor(topic))) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~ topic, scales = "free") +
  coord_flip()
```

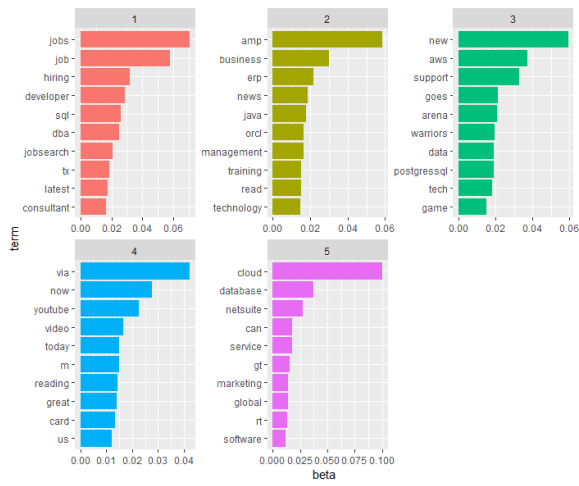



Figure 52 – Word-topic probabilities for Oracle

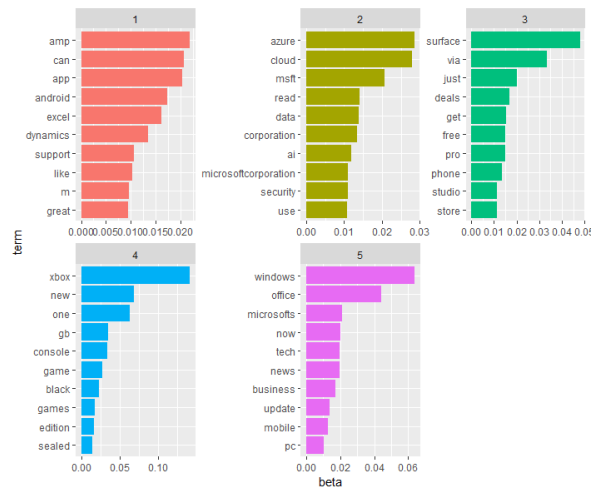


Figure 53 – Word-topic probabilities for Microsoft

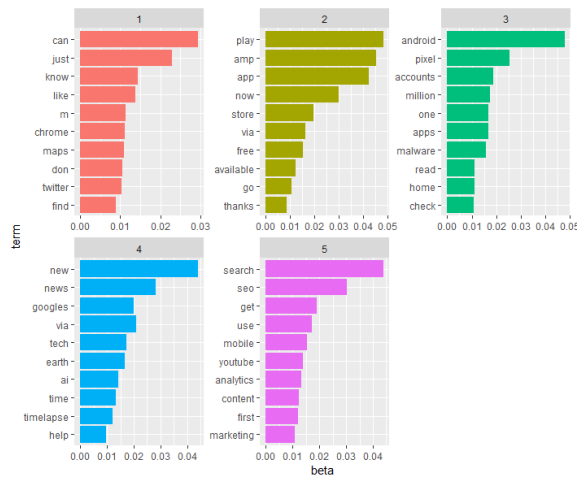


Figure 54 – Word-topic probabilities for Google

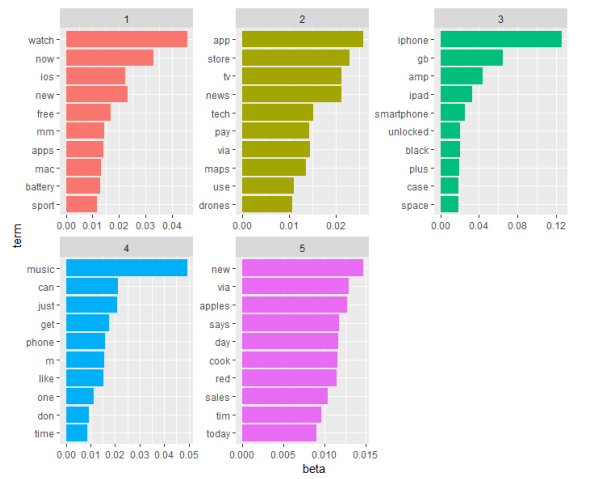


Figure 55 – Word-topic probabilities for Apple



Figure 56 – Word-topic probabilities for Amazon

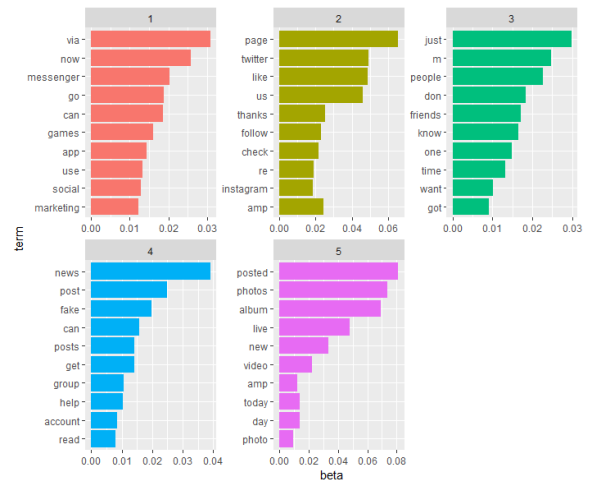


Figure 57 – Word-topic probabilities for Facebook

The LDA and word probabilities pass the coherence check, as all the topics and wording show semantic correlations. Additionally, the beta value shows the probability of the selected terms to be in a specific topic (per-term-per-topic).²⁶ In conclusion, none of the existing topics show similarities to stock market topics. To dig deeper and find stock market-related tweets, a lexicon-based approach will be executed. The idea is to build a list with a bunch of words for finance-related tokens and to compare it with the existing text. If the *text* contains any of the predefined tokens, a new logical variable named *stock_content* will contain the value 1, else it will be assigned as 0. A reference from the internet will be used, namely the Cambridge Dictionary, which provides stock-related tokens (Cambridge Dictionary, 2019). After extracting the data and transforming it into a csv file, the data got imported into R-Studio by executing the following script:

Script 13 – Building the variable *stock_content* by setting up a lexicon

```
stock.words <- read.csv("~/Master Project/stock words.csv", sep="")
View(stock.words)
stock.words <- as.list(as.data.frame(stock.words))
modify_final$stock_content <- as.factor(as.logical(sapply(stock.words, grepl, modify_final$text)))
view(modify_final)
```

Unfortunately, the tweets don't contain any significant amount of stock-related content, in fact, 0,002% were stock-related. Nevertheless, the variable was set up, as the objective of this chapter is to retrieve as much information as possible.

4.2.5. Observations and resulting Actions

This chapter analyzes the AS-IS state of the existing data set. It views in a univariate analysis the continuous variables, checks the frequencies and distribution of categorical variables and examined the text variable by performing mainly the LDA algorithm, which showcased that each company has similar but also distinct topics from each other. The following table summarizes all the significant insights of Chapter 6 and will define next actions for the data set:

Table 5 – Summary of observations and associated modification actions

ID	Variable	Observation	Action
1	Stock data	The continuous characteristics of the stock price have no significance for the further analysis	Drop all the stock market variable with the exception of <i>increase</i>
2	<i>text</i>	It contains noise in form of duplicate data entries, like bots, spam, advertisements, etc.	Identify and delete spam messages and duplicates
3	<i>text</i>	Redundant content, as the observation points contain company names	Remove company names from the <i>text</i> variable
4	<i>text</i>	Set a new variable which classifies each observation point as spam	Add new variable <i>duplicate</i>
5	<i>text</i>	Perform sentiment analysis with the most frequent semantic tokens to identify new discriminators	Perform sentiment analysis

²⁶ The R documentation explains the beta value in great detail (Robinsion & Slige, 2019)

6	<i>time</i>	As time has too many levels, add a new variable which classifies the time if it is within the trading opening time	Add new variable <i>trading_hours</i>
7	<i>year</i>	The year is a factor variable with one level	Drop variable year
8	<i>company</i>	The company is a factor variable. To further measure statistical indicators, it should be transformed into a numerical variable	Add new variable <i>company_id</i>
9	<i>Time</i>	Extract the hour from the time stamp and declare a new variable to decrease the factor levels	Add new variable <i>hours</i>

4.3. DATA MODIFICATION AND FEATURE ENGINEERING

This chapter will focus on the project objective “Apply sentiment analysis to the data set to label their semantic behavior” and “Apply feature engineering, especially by indicating word-frequencies, presence, and relevance”. Furthermore, it will target the discovered anomaly of the duplicates and spam messages, to clean as many noises as possible. The target is to build a semantic coherent data set with a greater number of features by extracting metadata to describe the variable *text*, particularly with sentiment analysis.

4.3.1. Modification and Transformation of the Data Set

The first step is to drop all the original stock market variables, as they have no semantic and statistical significance for the further procedure, with the exception of the target variable *increase*. Before targeting noise and outliers, a sentiment analysis, by making usage of the functionality set of the package *qdap*, will be performed to include the polarity analysis of the outliers. The following figure describes the sentiment distribution of the data set:

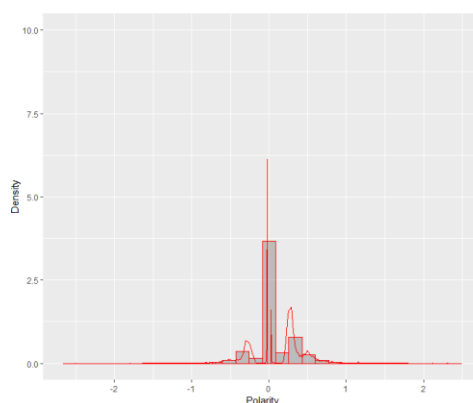


Figure 58 – Polarity histogram

To distinguish the tweets, which appear within the world stock exchanges opening time, another variable was set up, namely *trading_hours*. It will be declared as a logical variable, which contains the value TRUE if the tweet appears within the opening time. Also, two separate subsets will be set up to search for dissimilarities. The following code represents this logic:

```
final$trading_hours <- ifelse(final$hour > 8 & final$hour < 17, "1","0")
final$trading_hours <- as.factor(as.character(final$trading_hours))

### trading
trading <- final[which(final$trading_hours == "1"),]
### not trading
not_trading <- final[which(final$trading_hours == "0"),]
library(ggplot2)

ggplot(final, aes(final$trading_hours, fill = as.factor(final$increase))) +
  geom_bar(position = "fill") +
  facet_wrap(~final$date, ncol = 6) +
  xlab("Trading Hours") +
  ylab("Count") +
  guides(fill=guide_legend(title="Increase"))
```

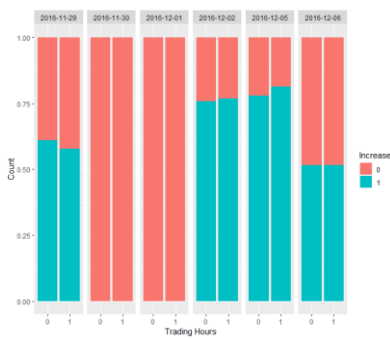


Figure 59 – Proportion of increased values by subsetting trading hours

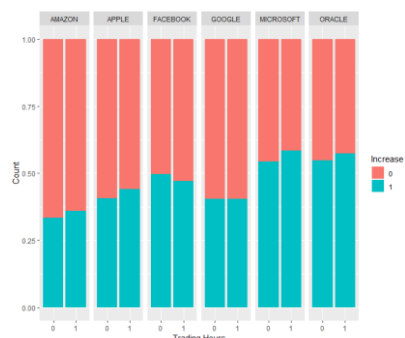


Figure 60 – Proportion of increased values by subsetting companies

A final modification, which will enhance the phase of spam recognition, is to set up the variable *duplicate*. This variable checks if the tweet content occurs once or several times. This logical variable will help to distinguish between spambots and unique tweets.

4.3.1.1. Experiment to classify unique tweets

Unique tweets are defined as text, which contains a strong opinion about a certain company. Its purpose should not be to promote a company, instead it should assess the company in a positive or negative way. The first step is to subset the final data set, the second step is to transform the variable text into a corpus, to parse it as a document term matrix. This document will be transformed into a data frame and will be merged with the target variable duplicate. After this step the data will be split in a training and in a test set to perform the classification and regression tree (CART). 99% of the highest word counts are used as predictor variables.

```
#building subsets
oracle_outliers <- modify_final[,c(1,9,11),]
oracle_outliers <- oracle_outliers[which(modify_final$company == "ORACLE"),]
oracle_experiment <- modify_final[which(modify_final$company == "ORACLE"),]

#Declaring corpus
corpus_oracle <- Corpus(VectorSource(oracle_experiment$text))
corpus_oracle <- tm_map(corpus_oracle, removeWords, stopwords(kind = 'en'))

#transforming DocumentTermMatrix
mydtm <- DocumentTermMatrix(corpus_oracle)
findFreqTerms(mydtm, lowfreq = 120)

#Sparse DocumentTermMatrix
sparse <- removeSparseTerms(mydtm, 0.99)
sparse

#Merging Dependent Variable
outliersSparse <- as.data.frame(as.matrix(sparse))
outliersSparse$duplicate <- oracle_outliers$duplicate
head(outliersSparse)

#Splitting Training and Test
set.seed(123)
n = nrow(outliersSparse)
index <- sample(n, n * 0.7)
train <- outliersSparse[index,]
test <- outliersSparse[-index,]

#Train CART
outliersCART <- rpart(duplicate ~ ., data=train, method="class")
summary(outliersCART)
prp(outliersCART)

#predict CART
predictCART <- predict(outliersCART, newdata=test, type = "class")
table(test$duplicate, predictCART)

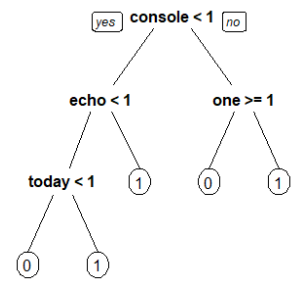
#overall accuracy (FP + TP) / (TF + TP + FP + TP)
```

This script trained a CART algorithm, which classifies spam messages with a moderate accuracy of ~76% for the company Oracle. This result is moderate, so this procedure will be applied for all the companies.

Company	Accuracy	Confusion Matrix	Classification Tree									
Oracle	76,3%	<table border="1"> <tr> <td></td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>2995</td> <td>216</td> </tr> <tr> <td>1</td> <td>886</td> <td>484</td> </tr> </table>		0	1	0	2995	216	1	886	484	
	0	1										
0	2995	216										
1	886	484										
Facebook	87.9%	<table border="1"> <tr> <td></td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>42138</td> <td>517</td> </tr> <tr> <td>1</td> <td>20038</td> <td>107410</td> </tr> </table>		0	1	0	42138	517	1	20038	107410	
	0	1										
0	42138	517										
1	20038	107410										

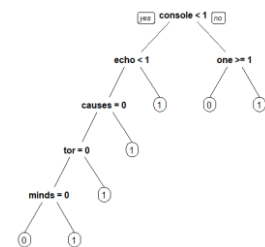
Microsoft 64,65%

```
      0  1
0 5458 207
1 3317 989
```



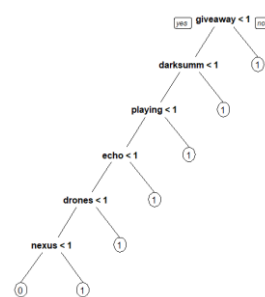
Amazon 66,9%

```
      0  1
0 7043 148
1 3975 1298
```



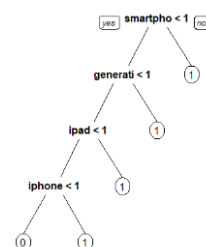
Google 67,8%

```
      0  1
0 24163 440
1 13854 5969
```



Apple 66,5%

```
      0  1
0 34940 10454
1 19549 25036
```



The results show an overall accuracy of ~65%. Facebook has a moderate signal to predict spam messages. This is due to the reason, that most of its duplicate values contain the update about photo uploads, which gets posted automatically on Twitter. For the further procedure, the variables assigned as duplicates will be dropped.

4.3.1.2. Sentiment Analysis with qdap and tidy

The sentiment analysis will be performed by using two different libraries, namely *qdap* and *tidytext*. Both libraries contain effective sentiment analysis function sets (Silge & Robinson, 2018). The library *qdap* quantifies the sentiment, whereas the *tidytext* package classifies the text in positive and negative sentiments, either with labels or remunerations (Silge & Robinson, 2017). The following script represents the *qdap* function, which will be applied to the data set. The result is the numeric

representation of the polarity.²⁷ Rinker (2019) describes in his qdap documentation that qdap needs the data to be cleaned from stopwords, and to be transformed into a vector character values. Qdap is known for being very accurate, but slow in terms of computational processing time. Indeed, the execution time for the following script took over 2 hours to execute. The equation of this algorithm assigns values to the sentence by tagging predefined dictionaries with a collection of weighted words and adds them up and divides them by n-identified word counts (Rinker, 2019).²⁸

Script 16 – Performing and merging the sentiment analysis output with the existing data set

```
#2 hours processing hours due to the large data set. Don't execute again
library(qdap)
library(dplyr)

sentiments <- qdap::polarity(final$text)
sentiments_df <- sentiments$all

#merge data sets
names(sentiments_df)[6]<-"text"
sentiments_final_df$ID <- seq.int(nrow(sentiments_final_df))
modify_final <- merge(modify, sentiments_final_df, "ID")
modify_final$text.y <- NULL
names(modify_final)[3]<-"text"
modify_final <- modify_final[,c(1,2,6,4,5,7,11,9,3,8,12,16,17,13,14,15,10)]

#plot sentiments
ggplot(modify_final, aes(x = modify_final$polarity, y = frequency(modify_final))) +
  geom_point(position = "jitter")

outliers_test$text <- as.character(outliers_test$text)
tidy_dataset <- outliers_test %>%
  unnest_tokens(word, outliers_test$text)

sentiments <- polarity(outliers_test$text)
sentiments <- data.frame(sentiments$all$polarity)

### plot the sentiments
ggplot(modify_final, aes(x = modify_final$polarity)) +
  geom_histogram(position="dodge") +
  xlab("Polarity") +
  ylab("Frequency")

ggplot(modify_final, aes(x = modify_final$polarity), fill=as.factor(modify_final$increase)) +
  geom_histogram(position="dodge") +
  facet_wrap(~modify_final$company) +
  xlab("Polarity") +
  ylab("Frequency") +
  ylim(0,50000)

ggplot(modify_final, aes(x = modify_final$polarity)) +
  geom_density() +
  facet_wrap(modify_final$company) +
  xlab("Polarity") +
  ylab("Density")
```

A large polarity document consisting of two lists is the result of this procedure, which results in a standard mean polarity of 0.23, and a standard deviation polarity of 0.29 of 18.406.600 word counts, which means that most of the tweets are classified as neutral with a tendency being classified as positive, with exception of Microsoft, as its density is distributed towards negative sentiments. As the

²⁷ The processing time of this script takes two hours to execute

²⁸ Rinker provides in his documentation a detailed summary about the equation and describes the algorithm in detail

current data includes a tremendous number of duplicates, it is assumed, that this fact skews the polarity towards neutralism. The following figures underline this hypothesis.

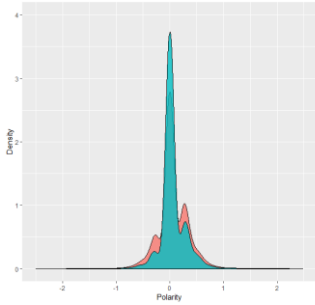


Figure 61 – Density of sentiment analysis with the package qdap

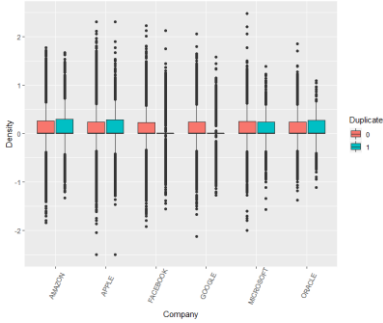


Figure 62 – Boxplots of sentiment analysis with the package qdap

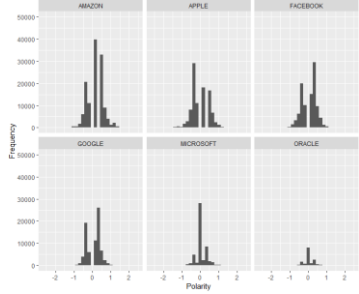


Figure 63 – Polarity histogram for each company

El-Din (2016) describes in a survey, that a key problem that sentiment analysis faces, is the variety of sentiment algorithms the market provides. The survey provides an overview of different methodologies for different semantic data. El-Din recommends for twitter sentiment analysis, lexicon-based approaches to precisely identify the correct tokens. For that reason, another sentiment analysis, which provides the package tidy, will be applied to the data set. Tidy contains a major amount of text mining applications and functions. Its sentiment analysis function set allows distinguishing between different approaches, namely afinn, Bing, nrc as well as loughran. They represent categorical sentiments by distinguishing text values as positive or negative, as well as numeric classification, where the sentiments will be renumerated within the ratio -5 and 5. For the further execution the lexicons nrc, Bing and afinn will be used.


```

#sentiment analysis with different libraries|
library(tidyr)
library(dplyr)
library(tidytext)
library(tidyverse)
library(tm)
install.packages("tm")

modify$text <- as.character(as.factor(modify$text))

tidy_sentiments <- Corpus(VectorSource(final$text))
tidy_sentiments <- tm_map(tidy_sentiments, removeWords, company_stopwords)

dtm1 <- TermDocumentMatrix(tidy_sentiments)
terms <- Terms(dtm1)
terms

ap_td <- tidy(dtm1)
ap_td

### library bing
ap_sentiments <- ap_td %>%
  inner_join(get_sentiments("bing"), by = c(term = "word"))

head(ap_sentiments)

##plot
ap_sentiments %>%
  count(sentiment, term, wt=count) %>%
  ungroup() %>%
  filter(n >= 3500) %>%
  mutate(n = ifelse(sentiment == "negative", -n, n)) %>%
  mutate(term, n, fill = sentiment) %>%
  ggplot(aes(term, n, fill = sentiment)) +
  geom_bar(stat = "identity") +
  ylab("Sentiments count") +
  coord_flip()

### library nrc
ap_sentiments <- ap_td %>%
  inner_join(get_sentiments("nrc"), by = c(term = "word"))

head(ap_sentiments)

##plot
ap_sentiments %>%
  count(sentiment, term, wt=count) %>%
  ungroup() %>%
  filter(n >= 7500) %>%
  mutate(n = ifelse(sentiment == "negative", -n, n)) %>%
  mutate(term, n, fill = sentiment) %>%
  ggplot(aes(term, n, fill = sentiment)) +
  geom_bar(stat = "identity") +
  ylab("Sentiments count") +
  coord_flip()

```

```

### library finn
ap_sentiments <- ap_td %>%
  inner_join(get_sentiments("afinn"), by = c(term = "word"))
head(ap_sentiments)

###plot
ap_sentiments %>%
  count(score, term, wt=count) %>%
  ungroup() %>%
  filter(n >= 5500) %>%
  mutate(n = ifelse(score == -3, -n, n)) %>%
  mutate(term, n, fill = score) %>%
  ggplot(aes(term, n, fill = as.factor(score))) +
  geom_bar(stat = "identity") +
  ylab("Sentiments count") +
  coord_flip() +
  guides(fill=guide_legend(title="Score"))

```

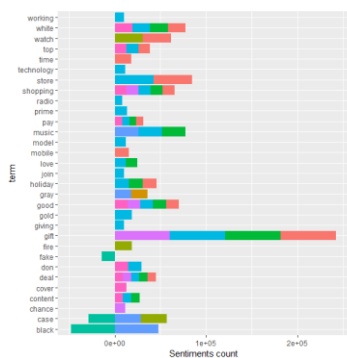


Figure 64 – Sentiments with nrc

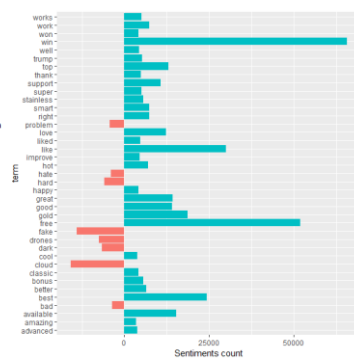


Figure 65 – Sentiments with bing

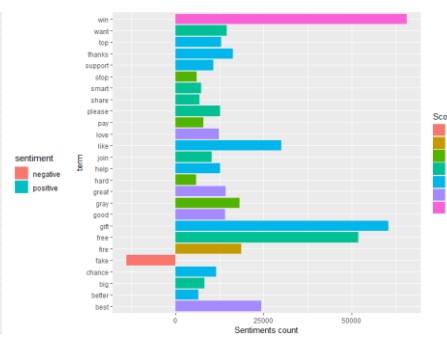


Figure 66 – sentiments with afinn

The figures 64, 65 and 66 represent the sentiments with different approaches. The lexicon-based approaches were not able to distinguish neutral words or words with a different meaning, for example, the token “cloud” is identified as a negative sentiment. Despite, it appears to provide the best classification amongst the other sentiment classifiers, thus it identified the largest amount of negative words. Respecting the curse of dimensionality, the decision was made to use the qdap method. The qdap approach will be merged into the data set with the unnest_tokens() function to turn this analysis to a one-row-per-term-per-document analysis, as Robinson (2019) showcases in his documentation.²⁹ Additionally, the character vectors within the variables *pos.words* and *neg.words* will be extracted and quantified to proceed with the analysis by using the following script:

²⁹ Script 17 includes the merge function

```
###clean pos and neg words plus quantity measures for pos and neg words
library(tidyr)
unique_unnest_1 <- unnest(modify_final, pos.words, .preserve = neg.words)
unique_unnest_2 <- unnest(unique_unnest_1, neg.words, .preserve = pos.words)
unique_unnest_2[, "pos.words.integer"] <- NA
unique_unnest_2$pos.words.integer <- strtoi(unique_unnest_2$pos.words, 35L)
unique_unnest_2[, "neg.words.integer"] <- NA
unique_unnest_2$neg.words.integer <- strtoi(unique_unnest_2$neg.words, 35L)
```

After that, the polarity analysis will be segmented into different groups of sentiments by binning the data set. This procedure will be applied manually, by setting up the ratios -2 as very negative, -1 as negative, 0 as neutral, 1 as positive, and finally 2 as very positive, which operationalizes the following script:

```
test$sentiment <- ifelse(test$polarity == 0, 0,
  ifelse(test$polarity > 0 & test$polarity < 2, 1,
    ifelse(test$polarity >= 2, 2,
      ifelse(test$polarity < 0 & test$polarity >
        -2, -1, -2))))
```

4.3.2. Final exploration and modification of the data set

After applying the adjustments in R, a final exploration and modification phase will allow to optimize the quality of the provided data set for the prediction phase. The final data set distinguishes between *duplicate* and *unique tweet* contents. As the previous word cloud shows, it includes noise due to the duplicates. The decision is made to delete the observations by identifying the duplicates with the binary variable *duplicate*. After dropping the duplicates, the observation points decreased drastically. It contains now 774.609 observation points, which makes the usage of different algorithms more efficient, as it decreases the computational power. By that the following word clouds and frequency tables highlight the most frequent positive and negative words:

to analyze differences in the model results. This experiment will be executed in the feature selection phase.³⁰

Table 6 – Final data set after the second cleaning phase for all companies with a set of 60 variables

Variable	Data type	Description
ID	Factor	Identification of observation point
date	Factor	Date of recorded tweet and stock price data
day	Factor	Numeric day of the date
dayname	Factor	Name of day
month	Factor	Month of date
time	Timestamp	Time of the recorded tweet
hour	Factor	Hour of the recorded tweet
company_ID	Factor	Identification of the company
company	Factor	Name of the company
text	Character	Content of the tweet
length_tweet	Integer	Length of the tweet content
duplicate	Logical	Distinguishes duplicates tweet as 1 and unique tweets as 0
pos.words	Factor	List of positive words included in the tweet content
neg.words	Factor	List of negative words included in the tweet content
pos.words.integer	Integer	Numeric transformation of pos.words character content
neg.words.integer	Integer	Numeric transformation of neg.words character content
wc	Integer	Count of registered words within the tweet
polarity	Numeric	Sentence polarity score (-3,...,3) -> (Bad,...,Good)
sentiments	Factor	Sets up sentiment ratios from polarity. -2 is very negative, -1 is negative, 0 is neutral, 1 is positive, and finally 2 as very positive
trading_hours	Factor	Filled as 1 if the observation point appeared within New Yorks trading opening hours, else 0
increase	Logical	Increase of stock market compared to the previous day is registered as 1, else 0
terms ³¹	Logical	Filled as 1 if the term exists in the variable text

4.4. MACHINE LEARNING CLASSIFICATION

This chapter will use Neural Networks to train a model to classify the target variable increase. It will examine the features, and it will apply statistical tests and machine learning algorithms to preselect important predictors. It will contribute to the project objectives “Predict the stock market fluctuation by targeting the variable increase” and “Analyze which keywords were highly relevant to classify the prediction”.

4.4.1. Model Selection for Binary Classification Problems

Many surveys were examined to find fitting machine learning algorithms for binary classification problems. Hsu, Chang, and Lin (2010) and other authors provide different methods and consult that Logistic Regression, Support Vector Machine and Neural Networks are effective procedures for binary classification models. It turns out, that especially Neural Networks are highly capable to

³⁰ Compare Figure in the Annex for the full list of data variables

³¹ In total >22 variables are selected for all companies. The company data set will include more variables.

predict binary targets. Brownlee (2019) state one common method is to use the sigmoid activation function, or cross-entropy error function, in a multiple-unit output layer within a Neural Network. Also, Brownlee showcases the success of this procedure by resulting classification problems with accuracies over 95% (Brownlee, 2019). Last but not least, epochs-driven deep learning methods are in general effective to find patterns within large data sets with limited sets of features.

A Logistic Regression model also can be trained for binary classification, and its field of use for classifications are widely applied in the research area and literature. Brownlee explains, that the logistic regression uses the sigmoid function, or also called the logistic function, which calculates the probability of the default class and transforms the probability with a sigmoid function into a classification (Brownlee, 2016) (Insitute for Digital Research and Education, 2019). Brownlee also explains important steps, which must be executed before the code will be applied. First the target variable has to be declared as logical, second logistic regressions are very prone to outliers, third “[i]t does assume a linear relationship between the input variables with the output” (Brownlee, 2016), fourth logistic regression assumes to drop correlated features, and fifth if the data input is very sparse it can fail to converge.

In the context of text mining, the literature includes many studies, which apply Support Vector Machines. The objective of this machine learning procedure is to calculate the hyperplane between n-populations. Usually this hyperlane is the outcome of n-supporting vectors, which are build based on the closest data points to calculate the so-called hyperlane (Drakos, 2018). Authors recommend to apply first the logistic regression, and if the problem is not linear separable, then to apply support vector machine with a non-linear kernel as Radial Basis Function (Drakos, 2018).

However, Singh (2018) provides a reference project for binary classifiers by training classification trees with a set of tokens, which successfully classifies spam messages, with an accuracy of over 90%. As the classification tree represents input variables as root notes and the output variables as the target variable it will be the simplest applied model to the data set. For splitting the decision nodes the algorithm chi-square will be applied. It is used for classification problems and for calculating the relationship between a set of categorical variables, though the categorical variables should not showcase any independencies as significant positive or negative correlations (Jain, 2017). When chi-squared is used as a splitting method the decision tree is called chi-square automatic interaction detector (Jain, 2017).

The following table summarizes in a cost-benefit analysis the pros and cons of each machine learning algorithm. It appears that the Neural Network is the most effective for the model development because of its high accuracy and automated feature engineering capabilities.

Table 7 – Cost-Benefit Analysis for selected machine learning models

Legend															
0=not good, 1=satisfactory, 3=good, 9=very good															
AS IS Data	Neural Network			Logistic Regression			CART			Support Vector Machine					
	positive	negative	Result	positive	negative	Result	positive	negative	Result	positive	negative	Result			
Mix of continuous and categorical variables		→ Only continuous, scaled and normalized input	● 3		→ Only categorical input	● 3	→ All inputs allowed	→ Can handle only small sets of factors	● 3		→ Only categorical input	● 1			
A high volume of observation points		→ Memory-intensive → High Time Complexity	● 1		→ Memory-intensive	● 1	→ Compared to other ML efficient		● 9		→ Memory-intensive	● 1			
Extracted Text features	→ High Accuracy	→ Mixed opinions about purpose of classification	● 9		→ Only categorical input	● 3	→ Results for text-based data are simple to understand and to compute		● 3	→ High Accuracy		● 9			
A set of limited and important features	→ Automated Feature Engineering	→ Difficult to understand results	● 9		→ Can handle only small sets of factors	● 1		→ Can handle only small sets of factors	● 3		→ Can handle only small sets of factors	● 1			
The dependent variable is categorical/ binary	→ Handles binary variable		● 9	→ Handles binary variable		● 9	→ Handles binary variable		● 9	→ Handles binary variable		● 9			
			31				17				27				21

4.4.2. Data Partition

The data partition phase is the preparation of the correct split of the data into training, validation and test partitions. An effective split is necessary to bypass underfitting and overfitting. As the names say, the training data set sets up the parameters of a model. Once the model is trained, the validation set validates the model by applying new data to it. Then the accuracies of the training and validation data set can be compared, to indicate any anomalies or significant changes within the results. Often the validation set is used to apply changes only in the parameter, and the third test data set is used to check the results. Many reference projects split their training and validation data set in 70% for training and 30% for test. Also, R provides functionalities, which splits this partition randomly. As the present data are addressing contents directed to specific companies a random split will be not selected, as this procedure can bias the data set. Although the existing contents are belonging to the same industry, one can argue, that a random split will not bias the data set. Chapter 6 and Chapter 8 though, clearly showcased that some terms are belonging only to one company. Further, a subset will help to produce a more efficient outcome for the models. Only the first 200.000 rows were selected. As the target variable is well distributed (60% increase; 40% decrease) it is not necessary to apply stratification methods. Also, Factors with 1 Level were excluded from the data set. Finally, to prevent overfitting and to optimize computational power a data sample was extracted from the original data set. The following R code operationalized this procedure:

```
#creating data sample, training and test set
sample1 <- sample_n(twitterNorm1, 10000)
str(sample1)
index <- createDataPartition(sample1$increase, p=0.7, list=FALSE)
df.training <- sample1[index,]
df.test <- sample1[-index,]
```

4.4.3. Feature Selection

The feature selection is the process, which enables to select the most relevant variables to allow an effective prediction. It is together with data cleaning one of the most important tasks, as a wrong set of predictors will have a negative impact on the accuracy. It allows decreasing the curse of dimensionality by redundancy reduction. Different techniques can be applied to check an effective selection of variables, namely reducing the redundancy by setting up the categorical independencies and continuous correlations with Pearson and Spearman correlations and checking the variable importance of selected machine learning algorithms (Shaikh, 2018). This chapter used many references from the Institute of Digital Research & Education, as they created a guideline with an overview of different approaches for feature selections (Insitute for Digital Research and Education, 2019).

First, the correlations will be checked to preview the signals and to drop redundant variables if necessary. The independencies of the categorical variables will be checked with a Chi-square statistical test. The higher the Chi-square value the more significant the variable is in terms of describing the target, as it measures the difference between the n-groups. The smaller the p-value, the higher is the significance of the predictions. It seems that the tokens for this procedure will apparently produce no relevant signal for the model. As the *date* and *day* are dependent, one of them will be dropped for the following procedure. The Pearson and Spearman correlation plots will help to identify positive and negative correlations within the continuous variables. The Institute of Digital Research and Technology recommends applying a Pearson correlation coefficient (2019). It seems that not a single variable showcases significant positive or negative correlations with the target *increase*. Only one term shows a positive correlation to the variable *increase*, with a Spearman coefficient of 0.28. Some terms are related, and it seems they appear dependently from each other. As a result, the tokens will be deleted.

Table 8 – Results of Chi-square test

Analysis	Contingency Table						Chi-square	P-Value	Importance
Increase ->	2016-11-29	2016-11-30	2016-12-01	2016-12-02	2016-12-05	2016-12-06	313520	2.2e-16	✓
date	0	38096	135341	126526	30674	31747			
	1	46861	0	0	85264	95277			

Script 21 – Applying Logistic Regression to analyze variable importance

```
fitall <- glm(increase~.,twitterDataNorm,family = "binomial")
varImp(fitall)
summary(step_fitall)
glm(formula = increase ~ day + month + company_number + length_tweet +
    wc + polarity + pos.words.integer + neg.words.integer + hour +
    trading_hours + sentiment + dayname_number + minute, family = "binomial",
    data = twitterDataNorm)
```

Script 22 – Applying Classification and Regression Trees to analyze variable importance

```
library(rpart)
cart <- rpart(increase ~ day + company_number + length_tweet + wc + pos.words.integer
    + neg.words.integer + hour + trading_hours + sentiment, data=df.training, method="class")
View(cart$variable.importance)
View(cart$p)
summary(cart)
str(cart)
plot(cart)
text(cart)
```

Script 23 – Applying Neural Networks to analyze variable importance

```
#Neural Network for rel. variable importance
set.seed(123)
nn14 <- neuralnet(increase ~day
    + company_number
    + length_tweet
    + wc
    + pos.words.integer
    + neg.words.integer
    + hour
    + trading_hours
    + sentiment,
    hidden = 3,
    data=df.training,
    err.fct = 'ce',
    likelihood = TRUE,
    stepmax = 1000000,
    linear.output = FALSE)

#variable importance
install.packages("devtools")
require(devtools)
source_gist('6206737')
gar.fun(mod.in = nn14, out.var = "increase")
```

Table 9 – Variable importance results of Logistic Regression

Variable	Variable Importance	P-value	Important
day	281.48	< 2e-16	✓
month	283.33	< 2e-16	✓
company_number	104.69	< 2e-16	✓
length_tweet	15.22	< 2e-16	✓
wc	7.98	6.61e-16	✓
polarity0.25	0.43	0.5961	-
polarity0.5	0.44	0.4678	-
polarity0.75	0.44	0.4724	-
polarity1	0.45	0.5029	-
pos_neg1	0.38	-	-
pos.words.integer	5.21	1.81e-07	✓
neg.words.integer	1.56	0.1170	-
hour	20.31	< 2e-16	✓
trading_hours	12.98	< 2e-16	✓
sentiment0.25	0.38	0.3384	-
sentiment0.5	5.46	4.88e-08	✓
dayname_number	325.89	< 2e-16	✓

Table 10 – Variable importance results of CART

Variable	Variable Importance	Important
day	88 ³²	✓
company_number	5125.84	✓
hour	1219.59	✓
wc	75.51	✓
trading_hours	56.53	✓
length_tweet	47.56	✓
pos.words.integer	29.02	✓
neg.words.integer	0.188	-
sentiment	0.11	-

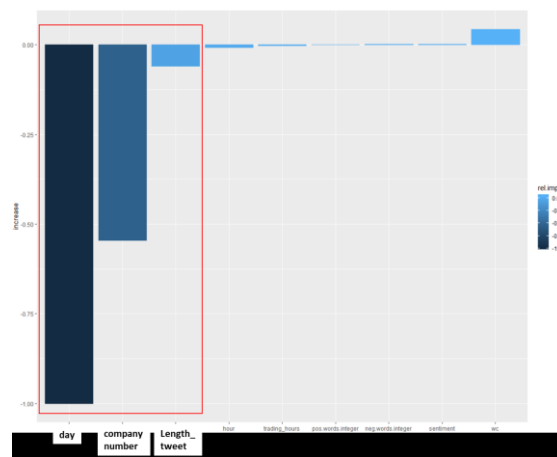


Figure 73 – Variable Importance results of Neural Network

All the models have in common that they did not hold any moderate importance or P-Values for the terms of polarities. Only the CART showcases significant variable importance values of positive classified terms. The CART was applied with different sets of variables, and as the Logistic Regression results that especially hour, and the date variables, together with the company labels, were highly important variables. Both results show, that when the variable *day* is used, the machine learning algorithms predict too well. Especially the CART algorithm predicted with 100% accuracy. For that reason, the variable *day* will be dropped for further analysis. Once the variable *day* was dropped, the accuracy of the CART algorithm dropped to 41,7%. All the described results will lead to the following actions: First, drop the tokens, as they have no significance for the independent variables and because of the existing bias. Next, drop the *wc* and *date* variable as it is correlated with other variables. Last, focus on the variables *day*, *hour*, *company*, *pos.words*, *neg.words*, and *length_tweet* as independent variables for the predictions. The *company_number* will be dropped as well, as the experiment focuses on the impact of the twitter features on the predictions.

³² This indicator was calculated in the first iteration of the experiment. The other indicators were calculated in the second iteration, as the variable *day* overfitted the model and had the highest variable importance of iteration 1.

4.4.4. Binary Classification with Neural Networks

On the search for a resource, which consults on machine learning for binary classification problems a survey from Campbell (2000), a survey from Zhang (2000) and a publication from Khan & Madden (2009) suggests using neural networks for binary classification problems. Also the source “Machine Learning in Automated Text Categorization” viewed, that Neural Networks are suitable to classify with moderate accuracy text and tokens (Sebastiani, 2001). Neural Networks require normalized data sets. It was decided to apply the Min-Max Normalization and to convert the variables, including the factor variables into continuous values (Drakos, 2018) (Gaur, n.d.). The following R-Code operationalized this procedure:

Script 24 – Converting and normalizing the variables

```
#Convert to numeric and apply normalization
library(dplyr)
twitter_data <- twitter_data %>% mutate_all(as.numeric)

#Normalization for neural networks
MinMax <- function(x){return((x- min(x)) / (max(x)-min(x)))}
twitterDataNorm <- data.frame(twitter_data$increase,
                             apply(twitter_data[,c(1:6,8:ncol(twitter_data))], MinMax))
```

After this transformation, the selected features were analyzed by using the library “GGpairs”. It allows to plot-important visualizations and metrics of the individual features. The following figure is the scatterplot matrix with the most significant features and the dependent variable:

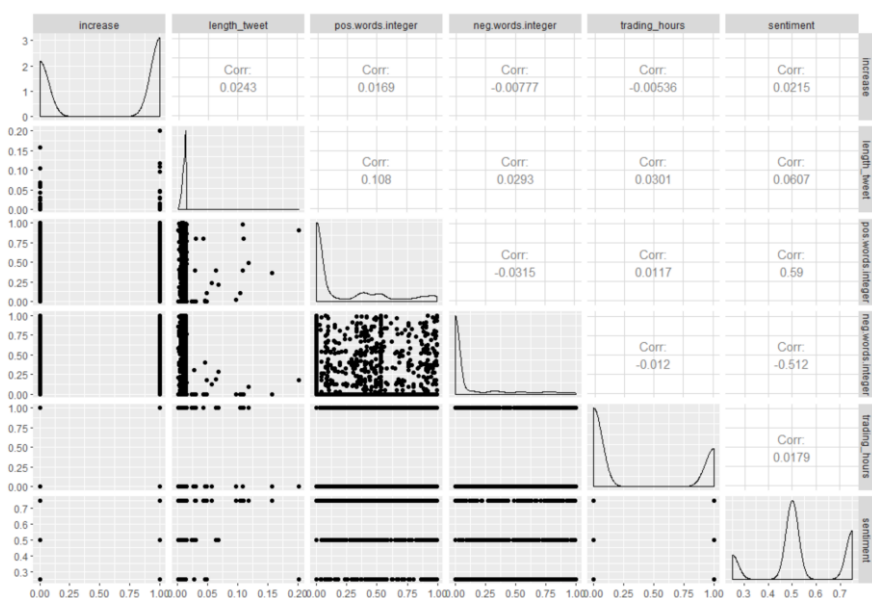


Figure 74 – Scatterplot Matrix for the used features

Furthermore, the library neuralnet provided the functionality set for the Neural Network application. One important parameter, which must be modified is the hidden layer. The literature review did not provide any standardized method, but Heaton (2017) explains one practical procedure, where he

highlights, that more than 2 hidden layers allow to train complex problems and find hidden signals within the data. It allows also to use the layers as “automated feature engineering” functions (Heaton, 2017). Further, it is important to decide how many neurons must be declared within the hidden layers to counteract underfitting and overfitting. Heaton explains: “[t]he numbers of hidden neurons should be between the size of the input layer and the size of the output layer. The number of hidden neurons should be $\frac{2}{3}$ the size of the input layer, plus the size of the output layer” (Heaton, 2017). Taking this procedure as a reference, the parameter for the following model was increased iteratively, starting with 2 hidden layers. Other authors explain carefully, that the higher the number of hidden layers and hidden neurons, the higher chance to overfit the model (Zhang, 2000). Important metrics as AIC, BIC, and Classification Errors were set up to choose the right Neural Network. The weights for these neurons though are essentially a black box and cannot be used for further analysis. As previously mentioned, by conducting iterations and experiments, the hidden layers were increased exponentially, namely 2 hidden layers, 2x2 hidden layers, 3x3 hidden layers, etc. Also, other parameters played an important role in this procedure. The error function was declared to cross-entropy. Cross-entropy is the loss function, which will be applied for the neural network, whose output value is between 0 and 1 (Han, Kamber, & Pei, 2012). The linear output was set up to false, as the output nodes should calculate the probability of an output belonging to a specific class. Last, the likelihood parameter will calculate the AIC and BIC. The AIC is the Akaike Information Criterion and the BIC is the Bayesian Information Criterion (Han, Kamber, & Pei, 2012). Both are important indicators for the model selection phase. The lower their value, the better the model build. The step max parameter was necessary to declare with a high number to allow the neural network function to use as many iterations as possible. In the end, many experiments were conducted by choosing a variety of features and by changing the parameters. The following R code provides the final parameter setting and features, which were used for the upcoming iterations. It appeared early, that R could not handle the computational power needed to succeed with the calculations. For that reason, the data samples were decreased.

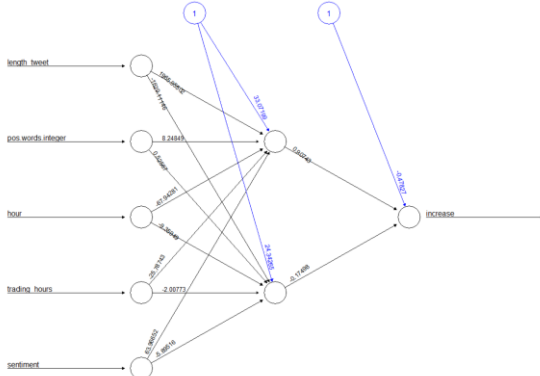
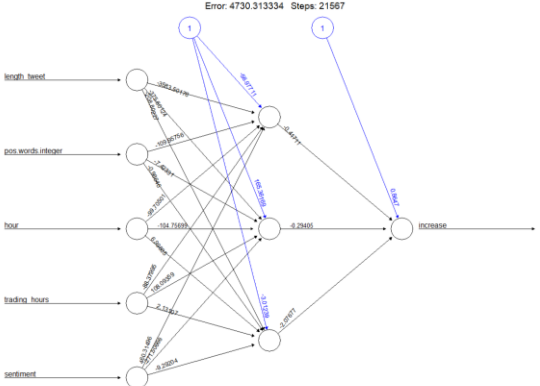
Script 25 – Training the first neural network

```
#Training first neural network
library(neuralnet)
set.seed(123)
nn1 <- neuralnet(increase ~length_tweet
                 + pos.words.integer
                 + hour
                 + trading_hours
                 + sentiment,
                 hidden = 4,
                 data=df.training,
                 err.fct = 'ce',
                 likelihood = TRUE,
                 stepmax = 10000000,
                 linear.output = FALSE)

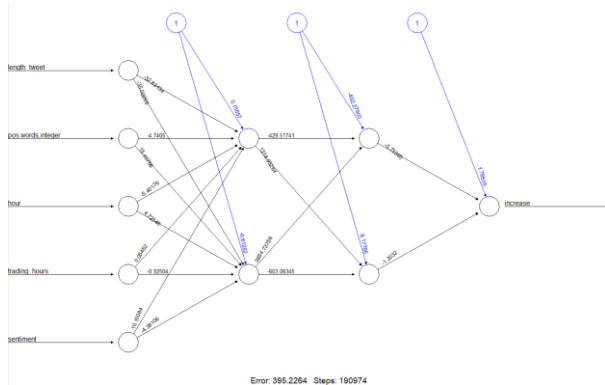
#Plot the results and predict with test set
nn1$result.matrix
plot(nn1)
summary(nn1)
nn1.results <- neuralnet::compute(nn1, df.test)
results <- data.frame(actual = df.test$increase,
                      prediction = nn1.results$net.result)
view(results)

#Confusion Matrix
roundedresults <- sapply(results,round,digits=0)
roundedresultsdf = data.frame(roundedresults)
attach(roundedresultsdf)
table(actual,prediction)
```

Table 11 – Results of Neural Networks

ID	Neural Network	Error	Cross-Validation	
			0	1
NN1		4730.32	0	1238
			1	1681
NN2		4717.76	0	1229
			1	1670

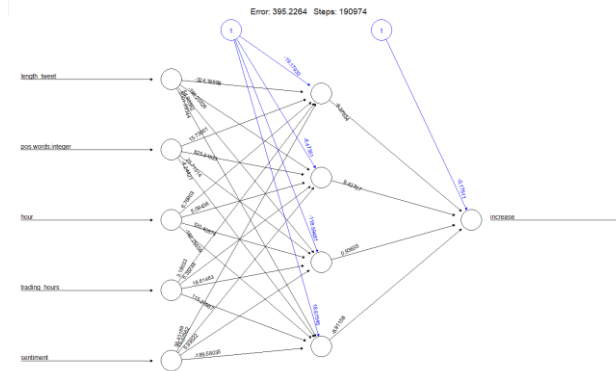
NN3



395.23

	0	1
0	95	79
1	142	83

NN4



378.87

	0	1
0	74	100
1	100	125

Error: 378.864404 Steps: 36557

4.4.5. Model Evaluation

As viewed before, the NN4 is the most accurate model, though it does not provide a moderate signal, as the accuracy demonstrates a result of ~54%. It appears that adding multiple hidden layers with multiple nodes is affecting negatively the confusion matrix and with it the accuracy. It is essential to calculate model selection metrics to measure the performance for all models. For that sake, the following statistical metrics will be calculated, namely i) Cross-Entropy Error, ii) AIC, as well as iii) BIC, iv) accuracy and v) rel. variable importance. The following indicators were calculated by sampling 3000 observation points from the original data set.

Script 26 – Calculating model selection metrics

```
#Confusion Matrix
roundedresults <- sapply(results,round,digits=0)
roundedresultsdf = data.frame(roundedresults)
attach(roundedresultsdf)
table(actual,prediction)

NN1_Train_Error <- nn14$result.matrix[1,1]
paste("CE Error: ", round(NN1_Train_Error, 3))

NN1_AIC <- nn14$result.matrix[4,1]
paste("AIC: ", round(NN1_AIC,3))

NN1_BIC <- nn14$result.matrix[5,1]
paste("BIC: ", round(NN1_BIC, 3))

#variable importance
install.packages("devtools")
require(devtools)
source_gist('6206737')

gar.fun(mod.in = nn14, out.var = "increase")
```

Table 12 – Model selection for 1 hidden layer with sample=3000

ID	Hidden Layer	Cross-Entropy Error	AIC	BIC	Test Set Accuracy
NL1	2	1206.44	2442.88	2525.32	57.4%
NL2	3	1198.98	2441.964	2562.97	57.3%
NL3	4	1194.99	2447.99	2607.381	58.3%
NL4	5	1181.923	2435.85	2633.705	56.4%

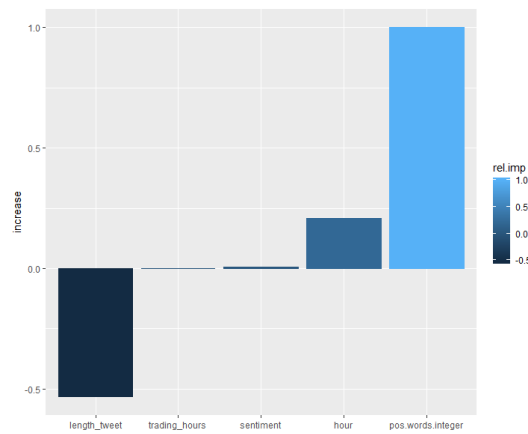


Figure 75 – Relative Variable Importance for final Neural Network

It appears that the Neural Network with 4 hidden layers has the best indicators for the present binary classification problem. The Github library 6206737 provided a library with a function set to calculate the relative variable importance for neural networks. The results explain that the *length_tweet* have the highest positive and the variable *pos.words* negative importance for the model. The last metric, which will be applied for this procedure is the Area under the Curve (AUC), which measures the performance for a classification problem by measuring the True-Positive-Rate and the False-Positive-Rates.

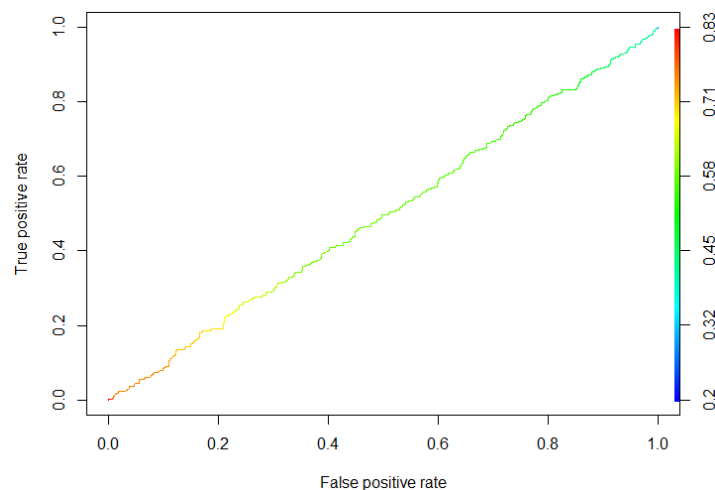


Figure 76 – ROC Curve for NNL4

It appears though, that the neural network, with this set of features, has no significant accuracy. In fact, the developed algorithm has an accuracy, which almost guesses the stock market fluctuation. To counteract this, the last experiment will be executed, by iterating through the feature selection by using the NNL4 model. The most significant signal will be used and iterated with a higher data sample.

Table 13 – Feature Selection Iterations with NNL4 model

Feature selection	Cross-Entropy Error	AIC	BIC	Test Set Accuracy
Extracted twitter features	1186.19	2430.39	2589.77	43.9%
Only stock related features	367.47	800.951	982.322	89.4%
Only important features	1151.22	2368.448	2549.82	59.1%

The table showcases, that when the stock related features, like *date*, *company_number*, etc. are included, the model performance increases up to 89%. The most significant features are *day*, *hour* and *company_number*. In the literature review it is described that the goal of model development is to generalize an equation, which helps to classify or predict an unseen observation point with moderate signal. Unfortunately, the features *day*, *hour* and *company_number* are context-specific, and the time dimensions will always be in the future. Also, its usage will obviously tend to overfit the model. The conclusion is, that the extracted features in this present project have no moderate significance for the classifier. The last attempt to optimize the model is to select the extracted twitter features and to increase the data sample. It seems that the accuracy increases up to ~60% and stagnates in this area. In the end, many experiments were conducted to reach a point with 60% accuracy. Unfortunately, the extracted features do not contain moderate signal to build an effective classifier with the preselected method Neural Networks.

Table 14 – Data sample Iterations with NNL4 model

Data sample	Cross-Entropy Error	AIC	BIC	Test Set Accuracy
5.000	1357.71	2781.43	2967.88	60.71%
10.000	4586.715	9239.429	9465.605	60.02%
20.000	9250.278	18566.556	18815.603	60.54%

5. CONCLUSION

Based on a twitter and stock market data set from 2006, merged from various American-based technology companies as well as yahoo finance API, this project analyzed the predictive capabilities a text-based variable can have on a binary target variable to forecast effectively the stock market fluctuation. Also, it described the AS-IS state of the existing data set by applying different text mining techniques and descriptive analytics. The research objective was to identify if tweets alone can have moderate signals to forecast the stock market movement without including important variables and metrics as recent studies explained (compare Nikfarjam, Emadzadeh, & Muthaiyah (2018)). Specifically tweets were targeted to build a set of categorical and continuous variables To extract as many features as possible, by respecting many authors suggestions, particularly to translate the text variables for the machine by transforming them into quantities and categories. The main technique for this method was natural language processing, text mining, as well as data mining, in particular sentiment analysis, CART, neural networks and logistic regression. To assure the outcome of this method a great time on data preprocessing and data cleaning was spend to identify spam and bot messages to clear the data set from noise and anomalies as insignificant tokens. In total over 1.700.000 million observation points were analyzed with statistical methods and complex figures were programmed to terminate the tweets discriminative power and to build a successive and iterative procedure, respecting the widely applied data mining framework CRISP by Shearer. As a tool the data science community R served a handy programming language to apply and to execute descriptive and predictive data mining methods. With a great number of powerful libraries and packages, the p-value of the programmed feature sets was analyzed by applying regression, Chi-square and logistic regression, classification trees and neural networks to identify features with discriminative power. With moderate importance the variables *hours*, *trading_hour*, *date*, *company_name*, *length_tweet*, *wc* as well as a *pos.words*, demonstrated moderate significance and were used to train the machine learning algorithms, in particular neural networks with different sets of hidden layers. The current results show no predictive capabilities. In fact the ROC calculated a result between ~40% and 60%. It is assumed, that the 1:n merge with the daily based stock market prices manifests an extreme bias within the data, which makes accurate predictions almost impossible. Another essential dispute is the fact that the tweets from various companies were aggregated into one data set. Respecting the insights from diverse researchers as Bollen, Mao, & Zeng, that text content and text features are context-sensitive, such research and projects must be performed using data and features only from one company to build an effective classifier.

5.1. ANSWER TO THE RESEARCH QUESTIONS

The present research project declared two research questions and a work breakdown structure, which structured the approach to provide a scientific relevant answer to the defined research questions. The following tables summarize the actions and results. After the table each research question will be answered.

ID	Project Task	Applied Techniques	Output
1	Summarize and review fundamental knowledge in the field of stock market prediction	Literature review	Work breakdown structure
2	Summarize and review fundamental knowledge in the field of text mining	Literature review	Collection of important R libraries
3	Preprocess the twitter data and perform data cleaning tasks to reduce noise	CMD, Outlier Analysis, Sentiment Analysis, Central Tendencies, Filter-methods	Twitter data sets were merged with stock market data sets. NAs were dropped and duplicates were excluded as they were skewing semantic content.
4	Apply text mining techniques to quantify the unstructured text into relevant features	Sentiment analysis with qdap, document frequency matrix	The polarity of the data set has a standard deviation with a slight tendency being positive. The LDA calculated product, release and advertisement-oriented topics. The most frequent positive and negative terms were extracted to include them in the data set.
5	Apply sentiment analysis to the data set to label their semantic behavior	Qdap and tidy Sentiment analysis	Tidy sentiments were not the right approach to classify the sentiments, as the majority of the tweets were classified into positive clusters. The qdap method showcased a well-distributed sentiment analysis. A coherency test was applied by classifying the quantities with thresholds and sub-setting the data set
6	Apply feature engineering, especially by indicating word-frequencies, presence, and relevancy	DTM, LDA, SA	Word-frequencies were extracted with a document term matrix and document frequency matrix. Due to limited computational power, only the top 20 positive and negative words were added to the data set.
7	Predict the stock market fluctuation by targeting the variable increase	Neural Networks, Logistic Regression, CART	With the features <i>hours</i> , <i>trading_hour</i> , <i>date</i> , <i>company_name</i> , <i>length_tweet</i> , <i>wc</i> as well as <i>pos.words</i> , and a parametrization of a neural network (4 Nodes, 1 Hidden Layer) it was possible to build a binary classifier with 60% accuracy.
8	Analyze which keywords were highly relevant to classify the prediction	Feature Selection	Unfortunately, no text-based features were relevant for the stock market predictions. Also, an experiment by sampling associated data from single companies to eliminate a possible bias was not resulting in moderate signals and results, but the accuracy increased slightly. The only text-based feature, which had importance was <i>length_tweet</i> and <i>pos.words</i> .

Research question 1: *What are the characteristics of the data set? How can descriptive approaches and features engineering support the model development, and is it possible to build a statistical signal for the model development?*

Originally, the hourly-based twitter data sets from 2016 included over 1.8 million observation points from the companies Amazon, Apple, Facebook, Google, Microsoft, and Oracle, which all belong to the information technology sector and were unevenly distributed. The selected API for the stock market prices were daily aggregated. This resulted in a radical character of the data set, namely the one-to-many relationship, directed from the features to the dependent variable increase. By that, the continuous variables from the stock prices were presenting a Pearson coefficient of 1 and were eventually deleted. The dependent variable increase presented the result of a stock drop or increase.

Early, a univariate and graphical analysis of the categorical variables highlighted outliers in the text length. As a result, the content of the subset was a plot with diverse word clouds, hierarchical clustering, frequency tables, which highlighted terms and tokens, which were referring to spam and bot messages. In fact, most of the observations were duplicates (~60%) with the highest proportion being Facebook. The duplicates further did not show any discriminative power to classify the target variable for any of the company data sets. Using Latent Dirichlet Allocation to analyze the content, and histograms to plot the frequencies, it became clear, that the duplicates skewed and biased the unique twitter terms, as the majority were referring to advertisement topics. Eventually, the duplicates were dropped for further analysis, which resulted in roughly 600.000 observation points.

On the other hand, the Latent Dirichlet Allocation showcased, with an optimal k-parameter being declared as 10, uniquely identified topics within all the companies, with the most common denominator being product presentations, job vacancies or release announcements. The topics overlapped but were showcasing unique wording and tokens for each company. Searching for semantic content, a sentiment analysis demonstrated strong opinions, both very positive comments as well as very negative comments. In total, the polarities have a well-distributed standard deviation, with a slight tendency being classified as positive. None of the categorical sentiment variables though were important discriminators, which was proved by applying a Chi-square statistical test and the variable importance of machine learning algorithms, as well as manual build graphical exploration insights. The filtered metadata of the time-stamp and the tweet content showcased significance for the model. The variables *day*, *company_name* and *hour* contained the highest P-value throughout all statistical tests and the highest variable importance. In fact, an experiment with a maximum depth

CART demonstrated, that a classification tree with maximum depth, could predict only by using day and hour the validation set with a moderate probability of roughly 90%, which obviously overfitted the classifier. Together with a set of positive and negative words and the variable tweet length, it was possible to gain a higher prediction rate (an increase from 40% to 60%).

In summary, the calculated features and the descriptive approaches supported the data cleaning and modification process, especially by identifying and dropping coherencies. Statistical tests though did not demonstrate any moderate significance for the predictions. The only important text-based features were *pos.words* and *length_tweet*. However, a neural network with 1 hidden layer and 4 hidden nodes allowed to build a binary classifier with 60% accuracy.

Research question 2: Which features, and keywords show influence on the stock market fluctuation?

Many experiments contributed to the feature engineering process, particularly sentiment analysis, tokenization, polarity analysis, document-frequency-matrix, and topic modeling. The modeling phase focused on the polarities, the term frequencies, as well as the presence of positive and negative terms. The p-value of the logistic regression was showcasing no significance. Also, classification trees were not able to make a split rule with positive and negative tokens, neither with their frequencies. A deep learning approach, namely neural networks, also did not calculate significant variable importance to the model. It appears that the metadata of the original variable *timestamp*, the filtered factor variable *day*, *company_name* and particularly the *hour* of the date had a great impact on all the models. The classification tree and the logistic regression demonstrated a high significance, which was observed also in the Chi-square test. Although the accuracy of the predictions was in the best case ~60%, it appears that such metadata are highly important for stock market predictions. These results also can be found in the mentioned study from the authors Kharb and Malik (2014) as well as Tsai and Hisao (2010). The authors identified, that time features demonstrate importance for stock price machine learning algorithms, which also underlines the hypothesis from Chavan & Patil (2013), namely that stock market predictions are time series analysis, and for that the time variables are extremely relevant to build an effective model.

5.2. LIMITATIONS AND FUTURE WORK

As previously mentioned, the limitation with the biggest impact on the present project is the fact, that multiple tweets were merged with daily-based stock market fluctuations. As data often represents itself in aggregated form, it can result in so-called ecological fallacy, once it is merged with individual data (Subarmanian, Jones, Kaddour, & Krieger, 2009). This phenomenon manifests when a data set inferences with an aggregated data set to which it belongs. Further, Subarmanian et al.

(2009) describe another statistical problem when individual observations are merged with aggregated data, namely the confusion of individual and ecological dependencies. The authors claim to drill the data to the closest denominator if possible. Contextualizing this, the present project should have aggregated the data by performing a daily-based time series analysis. Instead of highlighting the content of the text variable, the term frequencies could highlight, which positive and negative words appeared within the day. Then these features could be used to predict the stock fluctuation. A limitation, which prevented this procedure, was the limited-time series. Many projects were compared and in total all-time series, the analysis included long time periods, many times daily based for numerous years to train the model effectively. The simplest solution though would be if the stock market data was extracted as a daily based data set. A daily based data set, would allow targeting in real-time, if certain tweets had an effect on the stock prices. This would eliminate ecological fallacy and especially deep learning neural networks, would be able to make use of the text tokens tendencies and frequencies. For future work in this area, it is recommended to buy data sets from granular stock market databases or APIs to perform moderate analysis and to make use of deep learning neural networks to classify unseen twitter contents, as many authors recommend their performance for news-based content analysis. Further recommendations are to always include news-based content to stock market analysis projects. Although the present project had no moderate signal, neural networks were able to predict with ~60% accuracy the target variable by using only text-based tokens. Also, the metadata from tweets can demonstrate moderate signals, in particular by extracting the text length of the tweets. It must be highlighted, that, depending on the machine learning algorithm and the distribution of the text lengths, these variables must be scaled, especially when deep learning algorithms will be used. Recalling the statement from Stibel (2009) “[the] future, like any complex problem, has far too many variables to be predicted” can be considered as true. News-based content like tweets is only one parameter from a complex equation, as the project from Tsai & Hisao (2010) showcases. They used in their experiment 85 micro and macroeconomic variables to predict the stock market price. Without including news-based stock market predictions they reached a moderate accuracy with ~80%. A final recommendation for future projects in this research area is to apply similar manipulations, polarity analysis, and tokenization as this project, but to include micro and macroeconomic variables from the belonging industries and companies. Last, this project closes with the words from Wiliam Blake, a famous biologist from the 18th century, who conducted many experiments in his research area. Soon he understood that “[t]he true method of knowledge is experiment” (Blake, 1788) which applies for the unexplored area of news-based content predictions.

6. BIBLIOGRAPHY

- Aese, K. (2011).** Text Mining of News Articles for Stock Price Predictions. Trondheim, Norway: Norwegian University of Science and Technology
- Agrawal, A. (31.03.2017).** Logistic Regression Simplified. Retrieved 22.12.2018 from Medium: <https://medium.com/data-science-group-iitr/logistic-regression-simplified-9b4efe801389>
- Agyemang, M., Barker, K. & Alhaji, R. (2004).** Framework for Mining Web Content Outliers. Retrieved 05.01.2019 from Resarchgate: https://www.researchgate.net/profile/Ken_Barker/publication/220998785_Framework_for_mining_web_content_outliers/links/00b7d53298dad51994000000.pdf
- Bartsch, S. (2015).** Ein Referenzmodell zum Wertbeitrag der IT. Wiesbaden, Germany: Springer Verlag
- Beckmann, M. (2017).** Stock Prive Change Prediction using News Text Mining. Rio de Janeiro, Brazil: Insituto Alberto Luiz Coimbra de Pos-Graduacao e Pesquisa de Engenharia
- Beniot, K., Watanabe, K., Wang, H., Nulty, P. & Obeng, A. (12.30.2018).** Quantitative Analysis of Textual Data. Retrieved 11.10.2018 from Cran: <https://cran.r-project.org/web/packages/quanteda/index.html>
- Blake, W. (1788).** All Religions are one – The Approach of Doom. N.d.: N.d.
- Bollen, J., Mao, H. & Zeng, X. (2010).** Twitter mood predicts the stock market. Bloomington, USA: Indiana University
- Bouchet-Valat, M. (30.12.2018).** Snowball stemmers based on the C libstemmer UTF-8 library. Retrived 01.02.2019 from Cran: <https://cran.r-project.org/web/packages/SnowballC/index.html>
- Brownlee, J. (01.04.2016).** Logistic Regression for Machine Learning. Retrived 01.02.2019 from Machinelearningmastery: <https://machinelearningmastery.com/logistic-regression-for-machine-learning/>
- Brownlee, J. (19.05.2019).** How to Develop a Convolutional Neural Network to Classify Photos of Dogs and Cats (with 97% accuracy). Retrived 02.05.2019 from Machinelearningmastery: <https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/>
- Cambira, E., Das, D., Bandyopadhyay, S. & Feraco, A. (2017).** A Practical Guide to Sentiment Analysis. Cham, Switzerland: Springer International Publishing AG
- Camebridge Dictionary (20.05.2019).** *Stock Markets*. Retrived 20.07.2019 from Camebridge University Dictionary: <https://dictionary.cambridge.org/us/topics/finance/the-stock-market/>
- Campbell, C. (2000).** Kernel methods: A survey of current techniques. Bristol, England: Bristol University

- Chavan, P. & Patil, S. (2013).** Parameters for Stock Market Prediction. N.d., India: VIT Pune
- Drakos, G. (2018).** Support Vector Machine vs Logistic Regression. Retrieved 12.06.2019 from Towards Data Science: <https://towardsdatascience.com/support-vector-machine-vs-logistic-regression-94cc2975433f>
- El-Din, D. (2016).** A Survey on Sentiment Analysis Challenges. Retrieved 01.01.2019 from Researchgate: https://www.researchgate.net/publication/301649355_A_Survey_on_Sentiment_Analysis_Challenges
- Feinerer, I. (2008).** A Text Mining Framework in R and its Applications. Vienna, Austria: Vienna University of Economics and Business Administration
- Feldmann, R. & Sanger, J. (2007).** The Text Mining Book: Advanced Approaches in Analyzing Unstructured Data. New York, USA: Cambridge University Press
- Fellows, I. (30. 12 2018).** Package 'wordcloud'. Retrieved 17.11.2019 from Cran: <https://cran.r-project.org/web/packages/wordcloud/wordcloud.pdf>
- Galili, T. (30.12.2018).** Introduction to dendextend. Retrieved 22.12.2019 from Cran: <https://cran.r-project.org/web/packages/dendextend/vignettes/introduction.html>
- Gaur, P. (n.d.).** Neural Networks in Data Mining. Rajasthan, India: International Journal of Electronics and Computer Science Engineering
- Giudici, P. & Figini, S. (2009).** Applied Data Mining for Business and Industry. Chichester, UK: John Wiley & Sons LTD
- Goodrich, B., Kurkiewicz, D. & Rinker, T. (2018).** Bridging the Gap Between Qualitative Data and Quantitative Analysis. Retrieved 22.12.2019 from Cran: <https://cran.r-project.org/web/packages/qdap/index.html>
- Grosjean, P., Ibanez, F. & Etienne, M. (2018).** pastecs: Package for Analysis of Space-Time Ecological Series. Retrieved 23.12.2019 from cran.r-project.org: <https://cran.r-project.org/web/packages/pastecs/index.html>
- Grün, B. (2018).** Topic Models. Retrieved 02.01.2019 from Cran: <https://cran.r-project.org/web/packages/topicmodels/index.html>
- Han, J., Kamber, M. & Pei, J. (2012).** Data Mining – Concepts and Techniques. Waltham, USA: Elsevier Inclusive
- Heaton, J. (2017).** Hidden Layers. Retrieved 29.07.2019 from Heaton Research: <https://www.heatonresearch.com/2017/06/01/hidden-layers.html>
- Hsu, C., Chang, C. & Lin, C. (15.04.2010).** A Practical Guide to Support Vector Classification. Retrieved 06.09.2018 from Researchgate: https://www.researchgate.net/profile/Chenghai_Yang/publication/272039161_Evaluating_unsupervised_and_supervised_image_classification_methods_for_mapping_cotton_root_rot/

[links/55f2c57408ae0960a3897985/Evaluating-unsupervised-and-supervised-image-classification](https://stats.idre.ucla.edu/other/mult-pkg/whatstat/links/55f2c57408ae0960a3897985/Evaluating-unsupervised-and-supervised-image-classification)

- Hyun Duk Kim, M. G. (2013).** Ranking explanatory sentences for opinion summarization. N.d.: SIGIR
- Insitute for Digital Research and Education (16. 06 2019).** Hoosing the correct Statistical Test in SAS, STATA, SPSS and R. Retrieved 03.07.2019 from Insitute for Digital Research and Education: <https://stats.idre.ucla.edu/other/mult-pkg/whatstat/>
- Jadhav, K. & Wakode, M. S. (2017).** Sentiment Analysis of Twitter Data for Stock Market Prediction. International Journal of Advanced Research in Computer and Communication Engineering
- Jain, R. (20.03.2017).** Decision Tree It begins here. Retrieved 22.09.2019 from Medium: https://medium.com/@rishabhjain_22692/decision-trees-it-begins-here-93ff54ef134
- Kao, A. & Poteet, S. (2007).** Natural Language Processing and Text Mining. London, UK: Springer-Verlag
- Khan, S. & Madden, G. (2009).** A Survey of Recent Trends in One Class Classification. Gaillimh, Ireland: Springer Verlag
- Kharb, A. & Malik, M. (2014).** Behavioural Finance: An Impact of Investors Unpredictable Behavior On Stock Market. Rohtak, India: International Journal of Recent Development in Engineering and Technology
- Lattier, n.d. (24.02.2018).** *Your Guide to Latent Dirichlet Allocation*. Retrieved 13.04.2019 from Medium: <https://medium.com/@lettier/how-does-lda-work-ill-explain-using-emoji-108abf40fa7d>
- Leskovec, n.d, Rajaraman, n.d & Ullman, n.d, (13.04.2016).** Singular Value Decomposition. Retrieved 01.04.2019 from Stanford University: <https://www.youtube.com/watch?v=P5mlg91as1c>
- Linoff, G. & Berry, M. (2011).** Data Mining Techniques For Marketing, Sales, and Customer Relationship Management. Indianapolis, USA: Wiley Publishing Inclusive
- Liu, B. (2012).** Sentiment Analysis and Opinion Mining. Willston, Australia: Morgan & Claypool Publishers
- Lopes, M. (28. 03 2017).** Is LDA a dimensionality reduction technique or a classifier algorithm? Retrieved 24.04.2019 from Towards Data Science: <https://towardsdatascience.com/is-lda-a-dimensionality-reduction-technique-or-a-classifier-algorithm-eeed4de9953a>
- Mahmudova, S. (24.10.2018).** What is problems of Text Mining? Retrieved 02.10.2018 from Researchgate: https://www.researchgate.net/post/What_is_problems_of_Text_Mining
- McAllester, D., Srebro, N. & Urtasun, R. (2006).** Pattern Recognition and Machine Learning. N.d., USA: Springer Verlag
- Mejova, Y. (16.11.2009).** Sentiment Analysis: An Overview. Retrieved 16.01.2019 from Researchgate: https://www.researchgate.net/publication/264840229_Sentiment_Analysis_An_Overview

- Molnar, C. (12.04.2019).** Interpretable Machine Learning. Retrieved 11.05.2019 from Github.io:
<https://christophm.github.io/interpretable-ml-book/>
- n.d. (30.05.2019).** *Twitter API Docs*. Retrieved 01.08.2018 from Twitter Developer Docs:
<https://developer.twitter.com/en/docs.html>
- Neuwirth, E. (30.12.2018).** RColorBrewer: ColorBrewer Palettes. Retrieved 11.08.2018 from Cran:
<https://cran.r-project.org/web/packages/RColorBrewer/index.html>
- Nikfarjam, A., Emadzadeh, E. & Muthaiyah, S. (16.09.2018).** Text mining approaches for stock market prediction. Retrieved 30.09.2018 from IEEE:
<https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5451705>
- Oliveira, N., Cortez, P. & Areal, N. (2015).** The impact of microblogging data for stock market prediction: using Twitter to predict returns, volatility, trading volume and survey sentiment indices. Braga, Portugal: Algoritmi Centre
- Ponweiser, M. (02.05.2012).** Latent Dirichlet Allocation in R. Retrieved 14.03.2019 from University of Vienna: <http://epub.wu.ac.at/3558/1/main.y>
- Rinker, T. (02.06.2019).** Polarity Score (Sentiment Analysis). Retrieved 06.05.2019 from R Documentation:
<https://www.rdocumentation.org/packages/qdap/versions/2.3.2/topics/polarity>
- Rinker, T. (02.06.2019).** qdap Package Vignette. Retrieved 06.05.2019 from Github:
http://trinker.github.io/qdap/vignettes/qdap_vignette.html#cleaning
- Robinson, D. & Slige, J. (30.05.2019).** Tidiers for LDA objects from the topicmodels package Retrieved 06.05.2019 from Github:
https://juliasilge.github.io/tidytext/reference/lda_tidiers.html
- Robinson, D. (06. 01 2019).** Does Sentiment Analysis work? Retrieved 29.04.2019 from Variance Explained: <http://varianceexplained.org/r/yelp-sentiment/>
- Russia Today (02.04.2018).** April Fool? Tesla shares drop after Elon Musk's joke about bankruptcy. Retrieved 02.04.2018 from rt.com: <https://www.rt.com/business/422934-elon-musk-tesla-bankruptcy-joke/>
- Sebastiani, F. (2001).** Machine Learning in Automated Text Categorization. Ricerche, Italy: Consiglio Nazionale delle Ricerche
- Shaikh, R. (28.10.2018).** Feature Selection Techniques in Machine Learning with Python. Retrieved 02.11.2018 from Towards Data Science: <https://towardsdatascience.com/feature-selection-techniques-in-machine-learning-with-python-f24e7da3f36e>
- Shearer, C. (2000).** The CRISP-DM model: The new blueprint for data mining. Seattle, USA: Journal of Data Warehousing
- Silge, J. & Robinson, D. (2017).** Text Mining with R a Tidy Approach. n.d., USA: O'Reilly Media

- Silge, J. & Robinson, D. (30.12.2018).** Introduction to tidytext. Retrieved 25.09.2018 from Cran: <https://cran.r-project.org/web/packages/tidytext/vignettes/tidytext.html>
- Singh, M. (23.12.2018).** Text Mining and Natural Language Processing in R. Retrieved 23.12.2018 from Udemy: <https://www.udemy.com/text-mining-and-natural-language-processing-in-r/learn/v4/overview>
- Stibel, J. (22.01.2009).** Why We Can't Predict Financial Markets. Retrieved 05.08.2018 from Harvard Business Review: <https://hbr.org/2009/01/why-we-cant-predict-financial>
- Subarmanian, S., Jones, K., Kaddour, A. & Krieger, N. (2009).** Revisiting Robinson: The perils of individualistic and ecologic fallacy. Rockville, USA: US National Institute of Health
- Subhasree, B., & Groswami, S. (30.12.2018).** RSentiment: Analyse Sentiment of English Sentences. Retrieved 06.10.2018 from Cran: <https://cran.r-project.org/web/packages/RSentiment/index.html>
- Tsai, C. & Hisao, Y. (2010).** Combining multiple feature selection methods for stock prediction. Decision Support Systems. N.d., n.d.: Elsevier
- Viljamaa, S. (01.08.2017).** Improved Multivariate Outlier Removal in high volume IC Production Tests. Retrieved 06.12.2018 from Semantic Scholarship: <https://pdfs.semanticscholar.org/c89b/d19a444ae765d8b1e80aab2983e6546d6267.pdf>
- Wei, T., Simko, V., Levy, M., Xie, Y., Jin, Y. & Zemla, J. (30.12.2018).** Corrplot: Visualization of a Correlation Matrix. Retrieved 30.12.2018 from Cran: <https://cran.r-project.org/web/packages/corrplot/index.html>
- Wickham, H. (30.12.2018).** Package Plyr. Tools for Splitting, Applying and Combining Data: Retrieved 30.12.2018 from Cran: <https://cran.r-project.org/web/packages/plyr/plyr.pdf>
- Wickham, H., Francois, R., Henry, L. & Müller, K. (10.11.2018).** Dplyr: A Grammar of Data Manipulation. Retrieved 30.12.2018 from Cran: <https://cran.r-project.org/web/packages/dplyr/index.html>
- Wickham, H., Hester, J., Francois, R. & Jorgensen, M. (30.12.2018).** Readr: Read Rectangular Text Data. Retrieved 30.12.2018 from cran: <https://cran.r-project.org/web/packages/readr/index.html>
- Wickham, M., Chang, W., Henry, L., Pedersen, T., Takanashi, K., Wilke, C., & Woo, K. (12.12.2018).** Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics. Retrieved 30.12.2018 from Cran: <https://cran.r-project.org/web/packages/ggplot2/index.html> abgerufen
- Zhang, P. (2000).** Neural Networks for Classification. Georgia, USA: Georgia State University

7. ANNEX

Table 15 – Final data set

Variable	Data type	Description
ID	Factor	Identification of observation point
date	Factor	Date of recorded tweet and stock price data
day	Factor	Numeric day of date
dayname	Factor	Name of day
month	Factor	Month of date
time	Timestamp	Time of recorded tweet
hour	Factor	Hour of recorded tweet
company_ID	Factor	Identification of company
company	Factor	Name of company
text	Character	Content of tweet
length_tweet	Integer	Length of tweet content
duplicate	Logical	Distinguishes duplicates tweet as 1 and unique tweets as 0
pos.words	Factor	List of positive words included in the tweet content
neg.words	Factor	List of negative words included in the tweet content
pos.words.integer	Integer	Numeric transformation of pos.words character content
neg.words.integer	Integer	Numeric transformation of neg.words character content
wc	Integer	Count of registered words within the tweet
polarity	Numeric	Sentence polarity score (-3,...,3) -> (Bad,...,Good)
sentiments	Factor	Sets up sentiment ratios from polarity. -2 is very negative, -1 is negative, 0 is neutral, 1 is positive, and finally 2 as very positive
trading_hours	Factor	Filled as 1 if the observation point appeared within New Yorks trading opening hours, else 0
trading_hours	Factor	Filled as 1 if the observation point appeared within New Yorks trading opening hours, else 0
like	Logical	Filled as 1 if the variable name exists within the text
free	Logical	Filled as 1 if the variable name exists within the text
best	Logical	Filled as 1 if the variable name exists within the text
win	Logical	Filled as 1 if the variable name exists within the text
good	Logical	Filled as 1 if the variable name exists within the text
love	Logical	Filled as 1 if the variable name exists within the text
available	Logical	Filled as 1 if the variable name exists within the text
great	Logical	Filled as 1 if the variable name exists within the text
top	Logical	Filled as 1 if the variable name exists within the text
support	Logical	Filled as 1 if the variable name exists within the text
gold	Logical	Filled as 1 if the variable name exists within the text
work	Logical	Filled as 1 if the variable name exists within the text
right	Logical	Filled as 1 if the variable name exists within the text
better	Logical	Filled as 1 if the variable name exists within the text
trump	Logical	Filled as 1 if the variable name exists within the text
thank	Logical	Filled as 1 if the variable name exists within the text
well	Logical	Filled as 1 if the variable name exists within the text
smart	Logical	Filled as 1 if the variable name exists within the text
hot	Logical	Filled as 1 if the variable name exists within the text
won	Logical	Filled as 1 if the variable name exists within the text
cloud	Logical	Filled as 1 if the variable name exists within the text
fake	Logical	Filled as 1 if the variable name exists within the text
shit	Logical	Filled as 1 if the variable name exists within the text
drones	Logical	Filled as 1 if the variable name exists within the text
bad	Logical	Filled as 1 if the variable name exists within the text
hard	Logical	Filled as 1 if the variable name exists within the text
hate	Logical	Filled as 1 if the variable name exists within the text

problem	Logical	Filled as 1 if the variable name exists within the text
issue	Logical	Filled as 1 if the variable name exists within the text
issues	Logical	Filled as 1 if the variable name exists within the text
lost	Logical	Filled as 1 if the variable name exists within the text
fucking	Logical	Filled as 1 if the variable name exists within the text
wrong	Logical	Filled as 1 if the variable name exists within the text
miss	Logical	Filled as 1 if the variable name exists within the text
funny	Logical	Filled as 1 if the variable name exists within the text
sorry	Logical	Filled as 1 if the variable name exists within the text
damn	Logical	Filled as 1 if the variable name exists within the text
dead	Logical	Filled as 1 if the variable name exists within the text
problems	Logical	Filled as 1 if the variable name exists within the text
increase	Logical	Increase of stock market compared to the previous day are registered as 1, else 0
