Predicting the Stock Market Using News Sentiment Analysis

Majid Memari
Southern Illinois University Carbondale, memari@siu.edu

PREDICTING THE STOCK MARKET USING NEWS SENTIMENT ANALYSIS

by

Majid Memari

Bachelor of Industrial Engineering, Azad Tehran University, 2010

Master of Business Administration (MBA), Azad Qazvin University, 2015

A Thesis

Submitted in Partial Fulfillment of the Requirements for the

Master of Science Degree

Department of Computer Science

Graduate School Southern Illinois University Carbondale

May 2018

THESIS APPROVAL

PREDICTING THE STOCK MARKET USING NEWS SENTIMENT ANALYSIS

By

Majid Memari

A Thesis Submitted in Partial

Fulfillment of the Requirements

For the Degree of

Master of Science Degree

In the field of Computer Science

Approved by:

Dr. Shahram Rahimi, Chair

Dr. Norman Carver

Dr. Banafsheh Rekabdar

Graduate School

Southern Illinois University Carbondale

November 3rd, 2017

AN ABSTRACT OF THE THESIS OF

MAJID MEMARI, for the Masters of Science degree in Computer Science, presented on November 3rd, 2017 at Southern Illinois University, Carbondale, IL.

Title: PREDICTING THE STOCK MARKET USING NEWS SENTIMENT ANALYSIS

Major Professor: Dr. Norman Carver

Big data is a term for data sets that are so large or complex that traditional data processing application software is inadequate to deal with them. GDELT is the largest, most comprehensive, and highest resolution open database ever created. It is a platform that monitors the world's news media from nearly every corner of every country in print, broadcast, and web formats, in over 100 languages, every moment of every day that stretches all the way back to January 1st, 1979, and updates daily [1].

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield a significant profit. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on the newly revealed information thus are inherently unpredictable [2].

On the other hand, other studies show that it is predictable. The stock market prediction has been a long-time attractive topic and is extensively studied by researchers in different fields with numerous studies of the correlation between stock market fluctuations and different data sources derived from the historical data of world major stock indices or external information from social media and news [6].

The main objective of this research is to investigate the accuracy of predicting the unseen prices of the Dow Jones Industrial Average using information derived from GDELT database. Dow Jones Industrial Average (DJIA) is a stock market index, and one of several indices created by Wall Street Journal editor and Dow Jones & Company co-founder Charles Dow. This research is based on data sets of events from GDELT database and daily prices of the DJI from Yahoo Finance, all from March 2015 to October 2017. First, multiple different classification machine learning models are applied to the generated datasets and then also applied to multiple different Ensemble methods. In statistics and machine learning, Ensemble methods use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone. Afterwards, performances are evaluated for each model using the optimized parameters. Finally, experimental results show that using Ensemble methods has a significant (positive) impact on improving the prediction accuracy.

Keywords: Big Data, GDELT, Stock Market, Prediction, Dow Jones Index, Machine Learning, Ensemble Methods

ACKNOWLEDGEMENTS

In dedication to my thesis advisor Dr. Norman Carver, who has guided me these past few years. It is only because of him that I have reached this point in my journey and have attained the knowledge, resiliency, persistence, and support needed to accomplish my goal.

I would like to extend a personal thank you to Dr. Sharam Rahimi, who in conjunction with Dr. Norman Carver, assisted me continuously throughout my research.

I would like to acknowledge my committee members in addition to Dr. Norman Carver and Dr. Sharam Rahimi, Dr. Banafsheh Rekabdar who have taken the time to attend my defense.

Finally, I want to extend my appreciation to my family and friends without whom this accomplishment would not have been possible.

TABLE OF CONTENTS

CHAPTER PAGE

ABSTRACT i

ACKNOWLEDGMENTS iii

LIST OF TABLES vi

LIST OF FIGURES vii

1. INTRODUCTION 1

1.1 Statement of the Problem 2

1.2 Objectives and Contributions 3

2. BACKGROUND AND RESEARCH 5

2.1 Big Data 6

2.1A Misconceptions about Big Data 7

2.2 GDELT 8

2.3 Literature Review 13

3. METHODOLOGY AND EXPERIMENTS 16

3.1 Historical Data 16

3.2 Data Collecting 17

3.3 Model Selection 19

3.4 Preprocessing the Data 20

3.5 Time Lag 21

3.6 Feature Engineering 23

3.7 Classification 24

3.8 Dealing with Negative and Positive Sentiments 29

3.9 The Tools 30

3.10 Prediction.....30

4. THE RESULTS OF THE EXPERIMENTS 31

4.1 Classification 32

4.1.A Logistic Regression Classifier.....32

4.1.B K Nearest Neighbors Classifiers 35

4.1.C Support Vector Machine 36

4.1.D Random Forest Classifier.....38

4.1.E Multi-Layer Perceptron Classifier 42

4.2 Ensemble Methods 46

4.2.A Ensemble Voting Classifier 47

4.2.B Naïve Bayes Methods 48

4.2.C Bernoulli Naïve Bayes 49

4.2.D Adaptive Boosting Classifier.....49

4.2.E Ensemble Gradient Boosting Classifier 50

4.2.F Bagging Classifier 55

4.3 Summary of Classification Results 57

4.4 Prediction 58

5. CONCLUSION AND FUTURE PERSPECTIVE.....61

REFERENCES 63

APPENDIX A.....66

APPENDIX B 68

APPENDIX C 70

Vita 73

LIST OF TABLES

TABLE PAGE

Table 1 Quick Review of the GDELT Attributes 10

Table 2 List of Keywords for DJIA 18

Table 3 Feature Engineering 20

Table 4. Filtered and Unfiltered Data Sets 21

Table 5 The 30 Largest Countries by Net National Wealth 22

Table 6 Filtered Data Set Based on the Locations 23

Table 7 The average performance of the classification models Using Different Time Lags 25

Table 8 Negative and Positive News Sentiments 26

Table 9 Description for Parameters of Logistic Regression 32

Table 10 Description for Parameters of K Nearest Neighbors 34

Table 11 Description for Parameters of SVM 36

Table 12 Description for Parameters of Random Forest Classifier 38

Table 13 Description for Parameters of Multi-Layer Perceptron Classifier 41

Table 14 Ensemble Voting Classifier 46

Table 15 Descriptions for Parameters of Ensemble Voting Classifier 47

Table 16 Description for Parameters of Adaptive Boosting Classifier 49

Table 17 Description for Parameters of Gradient Boosting Classifier 50

Table 18 Description for Parameters of Ensemble Bagging Classifier 55

Table 19 Average Performance of the Classifiers 56

Table 20 Prediction Results Based on the Numbers of Samples in Each Data Set 57

Table 21 The Size of Training Set in Each Data Set 58

LIST OF FIGURES

FIGURE PAGE

Figure 1 DJI (Mar 01, 2015 - Oct 01, 2017) - Prices currency in USD.....28

CHAPTER 1

INTRODUCTION

Technology is one of the most influential aspects in modern history without which humankind would not be where they are today. By incorporating technology into everyday life, people have the opportunity to be more efficient and productive in every decision they make. One way that technology has been helpful is the topic of machine learning. Machine learning is a field of computer science that aims to give computers the ability to artificially learn [29]. It explores the study and construction of algorithms that can "learn" the patterns and make predictions on data [29]. Understanding the various techniques of machine learning enables individuals and entities to apply it to a wide range of areas. One specific area where these techniques have increasingly been applied to is the Stock Market prediction. Studies show that using machine learning algorithms such as Support Vector Machine (SVM) and Reinforcement Learning to predict stock market prices can make a huge impact on how and what individuals or businesses do with their money [4] [11] [13] [16] [17] [20]. These models have shown to be very effective in analyzing the stock market to optimize profits with a minimum amount of lash back [3] [20] [28]. Currently, in the United States, stock trades use special high-frequency trading algorithms that monitor the stock market [36] [37]. The problem with these kinds of machine learning models is that the inputs chosen are usually determined from the same dataset as the market that is being evaluated in the first place. However, modern-day technology has given entities the ability to share and use information outside and inside the market share. In fact, some studies show there is a strong correlation between closing prices overseas and its effect on the prices in the United States after they close. Therefore, it is necessary to evaluate information in a bigger perspective rather than keeping it the way it is currently because this makes the process of predicting future

prices more prone to flaws [28]. This flaw has been acknowledged and many efforts are being made to widen these boundaries to get the most accurate results possible. One way this is being done is through an approach known as sentiment analysis of up-to-date financial news and social networking streams such as Twitter [8] [23] [33]. Another way is using a database GDELT, which already has analyzed the sentimental analysis. This reduces the need to conduct a separate analysis and gather information from different sources.

1.1 Statement of the Problem

Two factors that affect the application of machine learning models to financial news are the ability to collect news and performing a sentiment analysis [10]. The common way to collect news is to run text mining [10]. This will aid in aggregating real-time news articles from various sources. Natural Language Processing (NLP) is then used to not only analyze the data from those news articles, but also to categorize those articles as positive (good) or negative (bad) polarity [10]. Optimizing the results derived from this analysis opens the ability to visualize the connection between news and its effect on stock market prices. With the assumption that negative and positive events over the course of a few days have an impact on the immediate future of stock market prices, it would be of great benefit to increase the accuracy of the prediction models so that individual and entities are able to choose the best course for their investments. GDELT has been chosen to drive this research because it addresses these both crucial factors. GDELT has the ability to collect news in more than sixty-five different languages from various streams nationally and globally and then runs a sentiment analysis on them [9]. By gathering relevant news and using the attributes to classify news as positive (good) or negative (bad), this research will aim to evaluate the accuracy of predicting the stocks prices [9]. The end

of this experiment should determine how and what relevant news affect stock prices and how this can be used to predict stock prices in the future.

1.2 Objective and Contributions

The objective of this study is to compare the performances of different machine learning algorithms that evaluate the relationships between equity, commodity, and currency markets with news sources and social network streams to aid in predicting the stock market prices more accurately. This study will also serve to show the value of inputting different features such as world major stock indices and social media trends to observe stock market's reactions. Therefore, the goal of this study is to provide an analysis on the different emerging algorithms and models (Logistic Regression, K Nearest Neighbors, SVM, Random Forest, Multi-layer Perceptron, Bernoulli Naive Bayes, Naive Bayes, Adaptive Boosting, Gradient Boosting, and Bagging) that are being applied to big data from stock market and other social media sources to determine there is a link between news and the Dow Jones Index and if it can be used to predict future stock prices. For the purpose of this study, the dataset used has been collected from GDELT. Future suggestions to expand this study would be to include GDELT V2.0, which uses near-real-time events and retrain models for high-frequency trading.

This study will be categorized as follows. Chapter 2 will consist of background information and research that have led to this study. This will include understanding big data, its misconceptions, GDELT, and other literature. Chapter 3 will continue to explain the experiment and methodology of this research. It will consist of how the data was collected and extracted, what features are used/selected, understanding the challenges, and finally determining a way to rid the noise and interpreting negative and positive sentiments. Other topics that will be covered in this experimental section are the tools being used to derive the information, how the

predictability will be determined, historical data and preprocessing the data. Chapter 4 consists of interpreting the data. It will cover all the models used and a description of the parameters used for each model. This chapter will conclude with a short summary of the results. Chapter 5 will discuss the results of the experiment, give a short overview of the entire paper, and will suggest future implications to consider for continuing experiments similar to this.

CHAPTER 2

BACKGROUND AND RESEARCH

The stock market is highly sensitive to many factors. Every minute of every day, it continuously fluctuates up and down. Part of the reason for this is due to supply and demand. When the need for a certain stock is high, its market price will increase. If more people want to sell a stock, in response the stock prices will decrease. There are several factors that affect this supply and demand relationship.

The types of news reports that are published at a certain given moment have an impact on how the stock prices react. Individuals are likely to sell stocks when informed of negative news that produces tension to sell and inadvertently cause a decrease in stock prices [9]. This amount of negative news is the result of poor corporate governance, bad earnings reports, unfortunate occurrences, and economic and political uncertainty [9].

On the other hand, positive news encourages individuals to purchase more stocks. Individuals are more likely to buy stock during this time due to increased buying pressure that leads to an increase in stock prices [9]. This amount of positive news is a reflection of innovation and acquisitions, good earnings reports, increased corporate governance, and an improvement in economic and political indicators [9].

Stock market prediction is vital and useful to all types of industries large or small. By being able to predict how the stock market will respond daily, investors can decide what to do with their money more efficiently. Stock prices are affected by the behaviors of investors whose behaviors are the direct result of publicly available information [18]. Therefore, financial news plays a crucial role in what individuals decide to do with their stocks [18]. There are various

theories evaluating the relationship between news articles and stock price movement but with poor accuracy [31]. The Efficient Market Hypothesis (EMH) and Random Walk theory state that stock prices are unpredictable and are driven solely by new information. Since new information is unpredictable, stock prices will move in a random walk pattern making it impossible to be used to predict future stock prices with an accuracy greater than 50 percent. Other studies show that stock prices do not follow a random pattern and instead can be anticipated regardless if the news gathered is unpredictable. A fair amount of studies shows that access to large amounts of data online such as through Twitter, Facebook, and LinkedIn can provide some insight in predicting changes in finances [14]. Outside of news, there seems to also be an existing link between views of Wikipedia on certain topics and the trends in the stock market, online chat activity and book sales, and even Google search queries and incidence of disease infection rates [21, 38, and 40]. In conjunction with this link, the idea of this research is to analyze the correlation between news and social media networks with future stock prices.

In this research, GDELT database is used to narrow down news media that makes references to the Dow Jones Index, which consists of the largest companies in the New York Stock Exchange, every day. The goal of this is to classify these news titles as positive or negative by based on their attributes to determine if there is indeed a link between the news reports and stock prices.

2.1 Big Data

Big data is neither a new term nor have its techniques never been used. In fact, big data has been around for years. Big data is data that is hard to interpret and expensive to manage. It is defined as high-volume, high-velocity, or high-variety information that needs to be processed in order to interpret and be used for purposes other than what it is originally intended for [7] [15].

Big data should meet any of or all four of the V's- volume, velocity, veracity, and variety [15]. There are two other V's less mentioned- variability and value [7] [15].

Volume refers to the size of the data, which needs to be considered to make sure it can be managed by the algorithm or models being used. Velocity focuses on the data streaming rates that need to be successfully handled by traditional algorithms. Veracity focuses on the idea that regardless of the data's availability, the quality of the data could be at risk. Larger amounts of data are prone to quality issues and need to be worked on before processing the information. Variety is the ability to present the same data in different ways and modalities while variability is the ability to change the structure of data and how it is interpreted. Lastly, value refers to the value that the data gives to the entirety.

2.1 Misconceptions about Big Data

There are common misconceptions about big data. The three most common concepts are that models are not important, correlation is enough, and older methodologies don't work anymore [35]. However, research and further evaluations show otherwise. Models exist because there is not just one simple model that would be perfect in every situation and result in the best performance and accuracy. Some studies on deep learning show that to interpret big data, certain sophisticated models are able to achieve better performance [35]. Deep learning techniques have been used for ages and simple models were more common in the past because there weren't any models equip to handle the data and parameters existing today. In addition, the amount of data available in past was significantly smaller than today, therefore making simpler models seem unnecessary today. However, this does not mean models should not be used at all. Complex models that exist today are very useful.

Regarding the second misconception of correlation not being enough, there are plenty of statistical information that shows that no amount of correlation data could ever replace the role of causality. For example, one study's data showed that there is a strong correlation between the number of hospitals and number of car thefts in that area. If the misconception that correlation is enough was true, then the best way to stop car thefts would be to stop making hospitals. However, this is not true [25]. There are other factors that could affect this correlation such as the location and economic status of the community. Therefore, it would not be valid to say that correlation is enough.

The third concept that old methodologies do not work anymore is also false. Big data has existed for many decades and it has been used many times in multiple different places. The only part of big data that has changed over time is its' volume. Therefore, methodologies about big data in the past should still be valued today.

2.2 GDELT

GDELT is also known as Global Database of Events, Location/Language, and Tone [8]. It monitors the world's broadcast, print, and web news from every corner of every country in over a hundred languages while identifying locations, people, organizations, themes, images, emotions, counts, and quotes every second of every moment [8]. GDELT is the largest and most comprehensive open-access spatiotemporal database in existence and is pushing the boundaries of "Big Data"[8]. A spatiotemporal database manages both space and time information [8]. For example, tracking moving objects at a specific position at a given time [8]. Creating such a platform that has information for every moment of everyday dating from the present back to January 1, 1979, has only been successful thanks to the technical and methodological innovations, partnerships, and creativity of those who worked together to make GDELT [8].

Nearly three-quarters of a trillion emotional snapshots and more than 1.5 billion location references were recorded in just 2015 [8]. Making GDELT this complex has required solving unparalleled challenges and “reimagining” human interaction and how it perceives societal-scale data.

Originally created by Yahoo! And Georgetown University, GDELT is machine-coded by the Textual Analysis by Augmented Replacement Instructions (TABARI) and is updated daily through information derived from thousands of news articles. It also uses Conflict and Mediation Event Observations (CAMEO) coding to record events [25]. CAMEO is a hierarchical coding set for recording events that are newsworthy and coverage [25]. It is commonly used in the study of political news and violence on a wide range of the spectrum. With all of this information, GDELT will encode the contents of each event into 57 different fields that help to describe that event, the people involved, and the geographical location along with other information [1] [11]. To understand GDELT, Actors need to be understood. Actor 1 refers to the one who's done something, and Actor 2 refers to the one the action is being done to. Each Actor is identified by a code, name, country code, country name, group, label, ethnic code and label, a religion code and label, and finally a CAMEO code and type [1]. There are more than 38 million multilingual news reports existent in the past 25 years that have been processed and extracted [11]. GDELT can be downloaded as a CSV file from its website or can be accessed through Google Big Query [1] [11]. Based on the available data, from the years 1979 to 2005 there are yearly archived files and from the years 2006 to March 2013, there are monthly archived files. After April 2013, there are daily archived files that include the web URLs of the recorded news events as well. More than 220,000 events are added to GDELT daily [1]. For the purpose of this study, ten features—"GLOBAL_EVENT_ID”, “EVENT_DATE”, “COUNTRY_CODE”, “IS_ROOT_EVENT”,

"SOURCE_URL", "NUM_MENTIONS", "NUM_SOURCES", "NUM_ARTICLES", "GOLDSTEIN_SCALE", and "AVG_TONE" have been chosen which are described in table 1. These attributes are chosen for this research because they are numerical attributes, which will help analyze the movements of stock prices.

Table 1

Quick Review of the GDELT Attributes [13]

GlobalEventID (integer)	A globally unique identifier assigned to each event record that uniquely identifies it in the master dataset.
Date (integer)	Date the event took place in YYYYMMDD format.
IsRootEvent (logical or binary or byte)	A binary number that shows if events occurring in the lead paragraph of a document tend to be the most "important." This flag can, therefore, be used as a proxy for the rough importance of an event to create subsets of the event stream.
GoldsteinScale (numeric)	Each event is assigned a numeric score from -10 to +10, capturing the theoretical potential impact that type of event will have on the stability of a country. This is known as the Goldstein Scale. This field specifies the Goldstein score for each event type. This score is based on the type of event, not the specifics of the actual event record being recorded – thus two riots, one with 10 people and one with 10,000, will both receive the same Goldstein score. This can be aggregated to various levels of time resolution to yield an approximation of the

Table 1

(Continued)

	stability of a location over time.
NumMentions (integer)	This is the total number of mentions of this event across all source documents. Multiple references to an event within a single document also contribute to this count. This can be used as a method of assessing the "importance" of an event: the more discussion of that event, the more likely it is to be significant. The total universe of source documents and the density of events within them vary over time, so it is recommended that this field is normalized by the average or another measure of the universe of events during the time period of interest. This field is updated over time if news articles published later discuss this event
NumSources (integer)	This is the total number of information sources containing one or more mentions of this event. This can be used as a method of assessing the "importance" of an event: the more discussion of that event, the more likely it is to be significant. The total universe of sources varies over time, so it is recommended that this field is normalized by the average or another measure of the universe of events during the time period of interest. Same as with NumMentions, this field is updated over time to reflect subsequent coverage of the event.

Table 1

(Continued)

NumArticles (integer)	This is the total number of source documents containing one or more mentions of this event. This can be used as a method of assessing the "importance" of an event: the more discussion of that event, the more likely it is to be significant. The total universe of source documents varies over time, so it is recommended that this field be normalized by the average or other measure of the universe of events during the time period of interest. Same as with NumMentions, this field is updated over time to reflect subsequent coverage of the event.
AvgTone (numeric)	This is the average "tone", general feelings or attitudes, of all documents containing one or more mentions of this event. The score ranges from -100 (extremely negative) to +100 (extremely positive). However, common values range between -10 and +10, with 0 indicating neutral. This can be used as a method of filtering the "context" of events as a subtle measure of the importance of an event and as a proxy for the "impact" of that event.

Table 1

(Continued)

SOURCEURL (character)	This field is only present in the daily event stream files beginning April 1, 2013 and lists the URL of the news article the event was found in. If the event was found in an article from the BBC Monitoring
ActionGeo_FullName (character)	This is the United Nations Country Codes of the matched location. This can be used to label locations when placing events on a map.

2.3 Literature Review

There are many types of research that have focused on the relation of news collected and its' influence on prices. To begin, it is worth noticing that there are a couple of famous sentiment analysis tools that are frequently used to process information. TextBlob is a type of natural language processing NLP that helps to provide a simple application program interface (API) that allows the completion of tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, and translation [12]. TextBlob uses NLTK (Natural Language Toolkit), a type of natural language processing library in Python. It contains two sentiment analysis modules- PatternAnalyzer and NaiveBayesAnaylizer [12]. PatternAnalyzer is based on the pattern library and returns the sentiment polarity and the subjectivity of the text [12]. The polarity score is usually within the ranges of [-1.0, 1.0]. The subjectivity is within the range of [0.0, 1.0] where 0.0 is very objective and 1.0 is very subjective. NaiveBayesAnalyzer is an

NLTK classifier that classifies movie reviews into positive or negative sets [12]. It is a text analyzer and returns the sentiment polarity of text.

A paper that has analyzed public sentiment to predict the movements of stock prices for thirty companies that are listed in the NASDAQ and the New York Stock Exchange uses NLP techniques to classify relevant news in the United States into five categories- Positive+, Positive, Neutral, Negative, and Negative- [34]. The hourly rise and fall of the Stock Market prices are normalized into five categories as well- Up+, Up, Flat, Down, and Down- [34]. The proposed algorithm is then used and discovers that there is, in fact, a link between the stocks of those thirty companies and the news with an accuracy of 76 percent [34]. Based on this research it can be determined that this technique has an average predictive accuracy better than NaiveBayes and SVM. Like this, there are plenty of research that seeks to find a correlation between stock prices and news and social media streams.

Another research uses GDELT to build a Hidden Markov Model that is used to predict the overall level of social unrest associated with country instability from five countries in the Southeast Asia- Thailand, Malaysia, Philippines, Indonesia, and Cambodia [24]. Social unrest event prediction was evaluated through calculations. Extensive empirical testing with the data specific to those five countries showed the effectiveness of using GDELT by comparing it with logistic regression model and the baseline model [24].

A recent experiment uses GDELT dataset to analyze the connection between China and the rest of the world. It first finds the total number of worldwide events that include China from years 1979 to 2012 and applies the ARIMA models to estimate future possible patterns in the years 2013 [17]. It seeks to determine the strength of correlation between China and the top 15

other countries from the list of possible events [17]. This study shows the effectiveness of using GDELT for predicting trends globally.

One study that runs parallel to this study tries to predict the Bitcoin prices using GDELT database [6]. Because the historical data for Bitcoin included the weekends, the time series in this research was complete and had no missing days. Therefore, the prices could be shifted to the next days. The time-lag between the event dates and the prices of the stocks adjust accordingly. This study evaluated prediction by only using the historical data for prices to future prices and then using machine-learning models along with historical prices to predict future stock prices [6]. This study concluded that the latter was more accurate.

Stock market prediction and big data are the main topics for this research. Although there is very little research on using GDELT solely to predict how news affects stock market prices, this research attempts to continue this fairly new topic and further evaluates its relationship to see if a correlation exists.

CHAPTER 3

METHODOLOGY AND EXPERIMENTS

The experiments done for this research consist of collecting financial news related to the DJI, a well-known index of the New York Stock Exchange, and then extracting certain features that influence the prices of the DJI. The movements of the DJI prices are needed to be learned by machine learning models in order to predict the movements of unseen prices. Since the desired outputs are either up or down, the binary classification machine learning models are needed to classify the data set in order to figure out whether there is an enough strong correlation between the relevant financial news and the DJI price movements, and if so, then to predict the movement of the DJI unseen prices. Binary or binomial classification is the task of classifying the elements of a given set into two groups based on a classification rule [10]. Some of the methods commonly used for binary classification are Decision Trees, Random Forests, Bayesian Networks, Support Vector Machines, Neural Networks, and Logistic Regression. Each classifier has its advantages in only a select domain based upon the number of observations, the dimensionality of the feature vector, the noise in the data and many other factors. Some tools to evaluate the performances of the models, as well as some technique to optimize them, are also needed such cross-validation and grid search. The goal of this experiment is to evaluate how the DJI prices change accordingly to relative news and see if there is an existing link between these two.

3.1 Historical Data

The historical data of the DJIA prices was downloaded from Yahoo Finance API in CSV format. The input data set includes the events from March 1st, 2015 to October 1st, 2017 (31

months) and the output data set includes the closing prices during this period excluding the weekends and holidays. The Figure 1 shows that the DJI prices have increased significantly in this period. As expected, there are more positive versus negative events for the news posted in this period especially in 2017.

Figure 1. DJI (Mar 1st, 2015 - Oct 01st, 2017) - Prices currency in USD from Yahoo Finance

3.2 Data Collecting

The first and foremost step in conducting this study is to successfully retrieve and gather data using GDELT for each day excluding weekends from March 1st, 2015 until October 1st, 2017 in CSV (comma-separated values) format files from the GDELT website [38]. The reasoning behind choosing this date range is because of the data before March 1st, 2015 seemed to have significantly more positive Average Tone in comparison to the events that occurred after that date. This might be due to another undocumented change that may have influenced GDELT's sentiment analysis tools before that time [6].

After collecting the data, CSV files were put together in chronological order. A table was created incorporating the features of GDELT using Oracle RDBMS, which is a relational database management system. Oracle RDBMS creates a platform on which a query can be created to search and select specific events that affect the DJI. The final CSV file consisting all the attributes from March 1st, 2015 until October 1st, 2017 was imported into the table.

Predicting a certain stock in the DJI is difficult because there may not be enough news related to a specific stock. It is because of this, an index of 30 companies of the Stock Market was chosen. The DJIA (Dow Jones Industrial Average) index was picked because it consists of the largest thirty companies in the NYSE [10] [19]. In addition, other indices consist of too many companies that make it very expensive to create a list of related keywords and keep track of all the news related to them. For example, the S&P index consists of 500 companies [10] [19].

A list of relevant keywords was created specifically to the companies in the DJI. This is because the DJI consists of only 30 companies and the best way to locate articles specific to these companies is through specific keywords pertaining to them. The list of the relevant keywords can be found in Table 2. The only feature available in GDELT database that can be used to search and find relevant news in regard to the query is “Source URL”. Uniform Resource Locator (URL), also known as a web address, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it [28]. Since URL’s contain news titles, searching through the “Source URL” column of the table is of benefit. Therefore, it was decided to search relevant keywords through the URL’s to find relevant news to the DJIA.

Table 2

List of Keywords for DJIA

Dow-Jones	DJIA	Industrial-Average	Jones-Industrial
^DJI	Dow-30	Charles-Dow	S&P-Dow-Jones
Apple-Inc.	American-Express	Boeing	Caterpillar
Cisco-Systems	Chevron	Coca-Cola	Disney
Dow-Du-Pont	Exxon	General-Electric	Goldman-Sachs
Home-Depo	IBM	Johnson-&-Johnson	Johnson-and-Johnson
JPMorgan-Chase	McDonald	Merck	Microsoft
Nike	Pfizer	Procter-&-Gamble	Procter-and-Gamble
Travelers-Companies	United Technologies	UnitedHealth	Verizon
Visa	Wal-Mart		

3.3 Model Selection

The first objective of this research is to find a meaningful correlation between the DJI related News Sentiments and the DJI price movements through specific machine learning models. The second objective of this research is to determine if there is indeed a way to predict the movements of the DJI unseen prices using the News Sentiments.

The performances of classification models have been analyzed using 10 Fold Cross Validation techniques. The K-Fold Cross Validation Score function repeatedly and randomly splits the training set into K-folds and then makes predictions based on each fold using a model

trained on the remaining folds [26]. In addition, the performance of each model has been evaluated using the Voting Classifier model from Ensemble method.

In order to do binary classification and considering the size of the data set, Logistic Regression, SVM, Random Forest Classifier, and KNN models were chosen for the experiments of this research.

The hyperparameters of the models which have significant influences on the performance of the classification models have been optimized by using the Grid Search function. This function exhaustively tries every combination of all the hyper-parameters for each model. In addition, the Ensemble Methods were used to improve the performance of the models, which combines the predictions of many base classifiers/estimators to improve generalizability over a single classifier/estimator.

3.4 Preprocessing the Data

Standardization of a data set is a common requirement for many machine learning models. They might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance) [32]. Therefore, for this study, all the numerical values have been standardized. To do this, the Standard Scaler function from preprocessing class of sklearn library is used. This function standardizes features by removing the mean and rescaling to unit variance [32]. In this research, centering and rescaling carried out independently on each feature by computing the relevant statistics on the samples in each set. The mean and standard deviation are then stored.

3.5 Time Lag

There is always some time lag between the date an event takes place and the date it affects the Stock Market because it takes time for an event to make an impact. One of the biggest challenges in this research is finding the average time lag between for all the events. It is determined that the DJI price movements would be shifted to the next business days to evaluate the best lag because this would give news ample time to make an impact, if it is going to, on prices.

The desired outputs (the movements of the DJI prices) have been shifted to the next working business day in the data. Each time the correlation between the News Sentiments and the shifted outputs is measured by applying them to different classification models and assessing how the results of classification generalize to n sample sets. APPENDIX B shows how this process was implemented. First, each time the desired outputs are shifted to the next day before reading the data, data = pd.read_csv('~/N-DAY-LAGGED.csv'). The input matrix consists of the News Sentiments, X = data [['AVG_GOLDSTEINSCALE', 'AVG_TONE']]. The desired output vector consists of the sifted the DJI price movements, y = data ['Price Change']. As you see in the following pseudocode, a for loop is created to break the input matrix and the output vector into n-sample sets and the inputs are standardized based on the samples of each set.

```
for i in range(0,646-n,1):
    Xi = X[i:(i+n)]
    yi = y[i:(i+n)]
    scaler = StandardScaler()
    Xi = scaler.fit_transform(Xi)
```

Then the samples of each set are split into train and test sets as shown in the following code snippet:

```
Xi_train,Xi_test,yi_train,yi_test=train_test_split(Xi,yi,test_size=m,random_state=None,shuffle=False)
```

The input and output train sets are fit to different classification machine learning models. Then the fitted classifiers predict the outputs of each test set based on their input test sets.

```
classifier = RandomForestClassifier()

classifier.fit(Xi_train,yi_train)

yi_predict = classifier.predict(Xi_test)
```

At the end, the average accuracy of the predictions is calculated by comparing the predicted outputs and the desired outputs as shown in the following snippet:

```
score = metrics.accuracy_score(yi_test,yi_pred)

scores.append(score)
```

The average accuracy of each classification model using different time lags was calculated and shown in the Table 7.

Table 7

The average score and the 95% confidence interval of average performance of the classification models Using Different Time Lags

Model	0 Day	1 Days	2 Days	3 Days	4 Days
Average Classification Performance	0.51+/- 0.01	0.52+/- 0.03	0.56+/- 0.02	0.53+/- 0.02	0.52+/- 0.03

As you see in Table 7, in average best classification performance we found was the result of fitting 2-day shifted outputs to the models. Therefore, we can claim that on average it takes 2 days for each event to have most influence possible in this data set on the DJI prices. For the rest of the research, we have used the 2-day shifted outputs.

3.6 Feature Engineering

In GDELT, records are completely independent of the past records. Given the enormous size of the data, it is not only very expensive to manage but also tedious to extract value from it [6]. Fitting all the GDELT features to predictive models increases the complexity of the models and as a result, decreases the overall performance.

To analyze the stock price movements in order to predict the movements of unseen prices some specific features need to be considered. The features “GLOBAL_EVENT_ID”, “EVENT_DATE”, “SOURCE_URL”, and “COUNTRY_CODE” can serve as unique identifiers, the numerical features “IS_ROOT_EVENT”, “NUM_MENTIONS”, “NUM_SOURCES”, “NUM_ARTICLES”, “GOLDSTEIN_SCALE”, and “AVG_TONE” can help to predict the

movements of the DJI prices. Each of these features has special capabilities that are essential in conducting this experiment. “GLOBAL_EVENT_ID” is used to help identify unique events, “EVENT_DATE” will be used to make the time series, and “SOURCE_URL” will be used find articles. In addition, “GOLDSTEIN_SCALE”, and “AVG_TONE” is specifically used in prediction and classification. “COUNTRY_CODE”, “IS_ROOT_EVENT”, “NUM_MENTIONS”, “NUM_SOURCES”, and “NUM_ARTICLES” have a special role in this experiment, they will be used to aid in filtering out irrelevant news or noise.

3.7 Classification

In statistics, dependence or association is any statistical relationship, whether causal or not, between two variables. Correlation most often refers to how close two variables have a relationship with each other. Correlations are useful because they can indicate a predictive relationship that can be exploited in practice. There should be a significant correlation between the input (the features of news) and 2-day shifted output (the price movement compared to previous day) of each day so that we can claim the movements of the DJI prices are predictable based of the DJI related news. To calculate the correlation, the input and output of each day fitted to the chosen classification models—Logistic Regression Classifier, K Nearest Neighbors, Random Forest, Multi-Layer Perceptron, Naive Bayes (Ensemble), Bernoulli Naive Bayes (Ensemble), Adaptive Boosting Classifier (Ensemble), Gradient Boosting Classifier (Ensemble), and Bagging Classifier (Ensemble). These models are most common classification models based on the studies surveyed for the purpose of binary classification.

One of the ways to evaluate the performance of classification models is to use the train-test sets techniques. Training the models against the smaller training set and evaluating them against the validation set may require a bit of work but the results are more accurate. However, by

partitioning the available data into three sets, we drastically reduce the number of samples which can be used for learning the model, and the results can depend on a particular random choice for the pair of (train, validation) sets. A solution to this problem is a procedure called the K-fold cross-validation. In K-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data. The k results from the folds are averaged to produce a single estimation. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once.

Therefore, K-fold cross-validation technique was used to evaluate the performance of each model, based on the size of the dataset (647 days), 10 folds were determined. The News Sentiments (Goldstein Scale and Average Tone) and all combinations of other features separately were fitted to different classification models. It was found that the classification models have best performances using only the AVG_AVGTONE and AVG_GOLDSTEIN_SCALE features as shown in Table 3. The average performance The News Sentiments (Goldstein Scale and Average Tone) and the other features separately were fitted to different classification models and their performances have been measured using 10-fold cross-validation technique. The average performance and 95% confidence interval of the classification models using only the News Sentiments and other features were calculated and shown in Table 3.

Table 3

Feature Engineering

Model	News Sentiments	Other features
Average Classification Performance	0.49 +/- 0.03	0.44 +/- 0.02

These results show that applying only the News Sentiments to the classification models results, in average, higher performance in comparison to applying other features. In addition, in general, using fewer features decreases the complexity of the classification models. Therefore, it was decided to apply only the News Sentiments to the models for the rest of the experiments. Usually, more important news has more influence on the stock market, to find relatively important news in the data set, the average values of all the numerical features for all the events occurred in each day were calculated. The unimportant news in dataset serves as noise, which decreases the performance of classification models. The numerical features "COUNT_URL", "AVG_NUM_MENTIONS", "AVG_NUM_SOURCES", and "AVG_NUM_ARTICLES" are chosen to filter out the unimportant news based on the average number of mentions, the average number of sources, and the average number of articles which are considered as thresholds, thus any event that has any feature value less than its threshold was removed from the dataset. Another feature used to filter out unimportant events is "Is Root Event". This feature shows whether the events occurring in the lead paragraph of a document tend to be the most "important." This flag can, therefore, be used as a proxy for the estimation of an event to create subsets of the event stream (GDELT). Any news that "is not root" have been filtered out. The

filtered and unfiltered data sets separately were applied to different classification models. The average performance and 95% confidence interval of the classification models using filtered and unfiltered dataset were calculated and shown in Table 4.

Table 4

Filtered and Unfiltered Data Sets

Model	Filtered dataset	Unfiltered dataset
Average Classification Performance	0.53 +/- 0.02	0.49 +/- 0.03

A considerable amount of important news occurs every day around the world but some of them do not affect the DJI prices. Therefore, it is very important to find the countries that relevant news have the most influence on the DJI companies. Some research has shown that the news related to the largest countries by net national wealth has the most influence on the largest companies in the NYSE, and the DJI consists of the 30 largest companies in the stock market [8]. Therefore, a list of the richest countries in the world is created and any events not pertaining to these countries have been ruled out with the use of the "COUNTRY_CODE" feature. The richest countries have been chosen based on Net National Wealth made by Credit Suisse Group in 2017. Credit Suisse Group is a Swiss multinational financial service holding company that operates the Credit Suisse Bank and other financial services investments. This list of countries can be found in Table 5. The average performance of the classification models was calculated to measure the correlation between events pertaining to the richest countries in the world. This can be found in Table 6.

Table 5

The 30 Largest Countries by Net National Wealth in 2017

1.United States	2.China	3.Japan	4.Unite Kingdom	5.Germany
6.France	7.Italy	8.Canada	9.Australia	10.South Korea
11.Spain	12.India	13.Switzerland	14.Tawiwan	15.Brazil
16.Russia	17.Netherlands	18.Belgium	19.Sweden	20.Mexio
21.Indoesia	22.Turrkey	23.Greece	24.Austria	25.Norway
26.Denmark	27.Singapore	28.Hong Kong	29.New Zealand	30.Israel

Table 6

Filtered data set based on the Locations

Model	The Richest Countries	Unfiltered Dataset
Average Classification Performance	0.56 +/- 0.02	0.53 +/- 0.02

Comparing the average of the classification performances using the News Sentiments related to the 30 richest countries to the average of the classification performances using all the News Sentiments shows that filtering the location of the News Sentiments helps in improving the classification performance.

3.8 Dealing with Negative and Positive Sentiments

We can fit negative and positive News Sentiments in two ways to the models. The first one is fitting the average of the negative and positive News Sentiments to the models and the second one is fitting negative and positive News Sentiments separately, this means fitting negative Goldstein Scale, negative Tone, positive Goldstein, and positive Tone separately to the models. There are no neutral values for the features Goldstein Scale and Average Tone. The average of News Sentiments, and positive and negative News Sentiments separately were applied to different classification models. The average performance and the 95% confidence interval of the classification models were calculated and the results are shown in Table 8.

Table 8

Negative and Positive News Sentiments

Model	Average	Separate
Average Classification Performance	0.56 +/- 0.02	0.53 +/- 0.02

Comparing the average of the classification performances using average News Sentiments to the average of the classification performances using negative and positive News Sentiments separately shows that using average News Sentiments helps more in improving the classification performance.

3.9 The Tools

Scikit-learn is a free software machine-learning library for the Python programming language [26]. It features various classifications, regression and clustering algorithms such as SVM, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy [27].

Scikit-learn was initially developed by David Cournapeau as a Google summer of code project in 2007 [27]. Its' name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy [27]. The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, and Vincent Michel, took leadership of the project and made its first public release on February 1st, 2010 [27]. As of 2017, scikit-learn is under active development [27].

3.10 Prediction

Having found the meaningful average classification performance of 0.67 we can claim that the movements of DJI prices are predictable. Clearly, the events of 31 months are not all dependent and the price movements of future days cannot be predicted using all the News Sentiments of the period. Therefore, the data set needs to be broken into smaller sets. To do so, a short period yet not too short that the machine learning models cannot learn the patterns, is most beneficial. To learn the patterns in each data set, there should be enough samples of two desired outputs (up or down). To train the predictive models, the News Sentiments and the desired outputs of train set of each sample set were used to train the predictive models. Having trained the models, the News Sentiments of the last 2 days of the sample set were applied to the trained models to predict, and the desired outputs of the last 2 days (test set) were compared to the

predicted outputs to test the performance of the predictions. Since the outputs were shifted to 2 next days we can consider the outputs of last 2 days of each sample set as future and unseen price movements. The process and results of the predictions are shown in the section 4.4.

CHAPTER 4

THE RESULTS OF THE EXPERIMENTS

4.1 Classification

The following are the results of the experiments conducted to get an accurate classification and determine if a significant correlation exists between the News Sentiments and the movements of the DJI prices between March 1st, 2015 and October 1st, 2017. The preprocessed data set, which is explained in the previous chapter, has first been fit to all the classification models and then their hyper-parameters were optimized to get the best accuracies.

4.1.A Logistic Regression Classifier

The first classification model used is Logistic Regression. Logistic Regression is a model where the dependent variable is categorical. It was first developed by David Cox, a statistician, in 1958. This classifier is used to estimate the probability of a binary response based on one or more predictor variables. The following parameters resulted in the best predictive accuracy for this data. The missing parameters have an insignificant effect on the overall accuracy, so their values have been made equal to their default values.

Best parameters of Logistic Regression for this data: C=1, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, max_iter=200, multi_class='ovr', n_jobs=1, penalty='l2', random_state=None, solver='newton-cg', tol=0.0001.

Performance in average: 0.54 +/- 0.03

Table 9

Descriptions of the Parameters of Logistic Regression

Paremeter	Description
penalty	Used to specify the norm used in the penalization. The ‘newton-cg’, ‘sag’ and ‘lbfgs’ solvers support only l2 penalties.
dual	Dual or primal formulation. Dual formulation is only implemented for l2 penalty with liblinear solver. Prefer dual=False when n_samples > n_features
tol	Tolerance for stopping criteria.
C	Inverse of regularization strength; must be a positive float. Like in support vector machines, smaller values specify stronger regularization.
fit_intercept	Specifies if a constant (a.k.a. bias or intercept) should be added to the decision function.
intercept_scaling	Useful only when the solver ‘liblinear’ is used and self.fit_intercept is set to True. In this case, x becomes [x, self.intercept_scaling], i.e. a “synthetic” feature with constant value equal to intercept_scaling is appended to the instance vector. The intercept becomes intercept_scaling * synthetic_feature_weight.

Table 9

(Continued)

class_weight	Weights associated with classes in the form {class_label: weight}. If not given, all classes are supposed to have weight one. The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y)).
solver	For small datasets, 'liblinear' is a good choice, whereas 'sag' and 'saga' are faster for large ones. For multiclass problems, only 'newton-cg', 'sag', 'saga' and 'lbfgs' handle multinomial loss; 'liblinear' is limited to one-versus-rest schemes. 'newton-cg', 'lbfgs' and 'sag' only handle L2 penalty, whereas 'liblinear' and 'saga' handle L1 penalty.
max_iter	Useful only for the newton-cg, sag and lbfgs solvers. Maximum number of iterations taken for the solvers to converge.
multi_class	Multiclass option can be either 'ovr' or 'multinomial'. If the option chosen is 'ovr', then a binary problem is fit for each label. Else the loss minimised is the multinomial loss fit across the entire probability distribution. Does not work for liblinear solver.

4.1.B K Nearest Neighbors Classifier

The second predictive classifier used is KNN. This is a non-parametric method that is used for classification and regression. The following parameters have been determined to get best predictive accuracy for this data. The missing parameters here are also insignificant that the values are considered to be equal to their default values.

Best parameters of K Neighbors Classifier for this data set : algorithm='auto', leaf_size=30, metric='minkowski', metric_params=None, n_jobs=1, n_neighbors=17, p=2, weights='uniform'

Performance in average: 0.56 +/- 0.02

Table 10

Descriptions for the parameters of the K Nearest Neighbors

Paremeter	Description
n_neighbors	Number of neighbors to use by default for kneighbors() queries
weights	weight function used in prediction
algorithm	‘auto’, ‘ball_tree’, ‘kd_tree’, ‘brute’
leaf_size	Leaf size passed to BallTree or KDTree. This can affect the speed of the construction and query, as well as the memory required to store the tree. The optimal value depends on the nature of the problem.
p	Power parameter for the Minkowski metric. When p = 1, this is equivalent to using manhattan_distance (l1), and euclidean_distance (l2) for p = 2. For arbitrary p, minkowski_distance (l_p) is used.

Table 10

(Continued)

metric	the distance metric to use for the tree. The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric.

4.1.C Support Vector Machine

The third predictive classifier used is SVM. Support Vector Machine are supervised learning models that work with learning algorithms that analyze data used for classification and regression. The SVM training algorithm builds a model that assigns new examples to the category [17]. The following parameters have resulted in the most accurate predictive data. The missing parameters are insignificant, so their values have been changed to their default values. SVM is based on libsvm [4] [17]. The fit time complexity is more than quadratic with the number of samples, therefore making it hard to scale to a dataset with more than a couple of 10000 samples [4]. The multiclass support is handled according to a one-vs-one scheme. The following is the optimum estimator for the dataset.

Best parameters of Support Vector Machine for this data set: C=1, cache_size=200, class_weight=None, coef0=0.0, _function_shape='ovr', degree=3, gamma=0.001, kernel='rbf', max_iter=-1, probability=False, random_state=None, shrinking=True, tol=0.001, verbose=False

Performance in average: 0.53 +/- 0.03

Table 11

Descriptions of the Parameters of the SVM

Parameter	Description
C	Penalty parameter C of the error term.
kernel	Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape (n_samples, n_samples).
degree	The degree of the polynomial kernel function ('poly'). Ignored by all other kernels.
gamma	Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. If gamma is 'auto' then 1/n_features will be used instead.
coef	Independent term in kernel function. It is only significant in 'poly' and 'sigmoid'.
probability	Whether to enable probability estimates. This must be enabled prior to calling fit, and will slow down that method.
shrinking	Whether to use the shrinking heuristic.
tol	Tolerance for stopping criterion.
cache_size	Specify the size of the kernel cache (in MB).

Table 11

(Continued)

class_weight	Set the parameter C of class i to class_weight[i]*C for SVC. If not given, all classes are supposed to have weight one. The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n_samples / (n_classes * np.bincount(y))
max_iter	Hard limit on iterations within solver, or -1 for no limit.
decision_function_shape	Whether to return a one-vs-rest ('ovr') decision function of shape (n_samples, n_classes) as all other classifiers, or the original one-vs-one ('ovo') decision function of libsvm which has shape (n_samples, n_classes * (n_classes - 1) / 2).

4.1.D Random Forest Classifier

The fourth predictive model used is the random forest classifier. This is a meta-estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and controls its' over-fitting [28]. The sub-sample size is always the same as the original input sample size but, the samples are drawn with replacement if bootstrap=True (default) [28]. The following parameters have resulted in the best predictive accuracy for this data. Like the other models, the missing parameters have little to no effects on the accuracy and therefore have been changed to their default values.

Best parameters of Random Forest Classifier for this data set: bootstrap=True, class_weight=None, criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=7, n_jobs=-1, oob_score=False.

Performance in average: 0.57 +/- 0.02

Table 12

Descriptions for the Parameters of the Random Forest Classifier

Parameter	Description
n_estimators	The number of trees in the forest.
criterion	The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain.
max_features	The number of features to consider when looking for the best split: If int, then consider max_features features at each split. If float, then max_features is a percentage and int(max_features * n_features) features are considered at each split. If “auto”, then max_features=sqrt(n_features). If “sqrt”, then max_features=sqrt(n_features) (same as “auto”). If “log2”, then max_features=log2(n_features). If None, then max_features=n_features.

Table 12

(Continued)

max_depth	The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. min_samples_split: int, float, optional (default=2) The minimum number of samples required to split an internal node: If int, then considers min_samples_split as the minimum number. If float, then min_samples_split is a percentage and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.
min_samples_leaf	The minimum number of samples required to be at a leaf node: If int, then considers min_samples_leaf as the minimum number. If float, then min_samples_leaf is a percentage and ceil (min_samples_leaf * n_samples) are the minimum number of samples for each node.
min_weight_fraction_leaf	The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

Table 12

(Continued)

max_leaf_nodes	Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as a relative reduction in impurity. If None then unlimited number of leaf nodes.
min_impurity_split	The threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise, it is a leaf.
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value. The weighted impurity decrease equation is the following: N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity) where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R, and N_t_L all refer to the weighted sum, if sample_weight is passed.
bootstrap	Whether bootstrap samples are used when building trees.
oob_score	Whether to use out-of-bag samples to estimate the generalization accuracy.

4.1.E Multi-Layer Perceptron Classifier

The fifth predictive model used for this research is the Multi-Layer Perceptron Classifier. This is a class of feedforward artificial neural networks. It consists of at least three layers of nodes. Each node, in except to the input node, is a neuron that uses a nonlinear activation function [21]. This model optimizes the log-loss function using LBFGS or stochastic gradient descent [21]. The following parameters result in the most accurate predictive results and the missing parameters are considered to be so insignificant that their values are changed to their default values.

Best parameters of MLP Classifier for this data set: activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(100,), learning_rate='invscaling', learning_rate_init=0.001, max_iter=200, momentum=0.9, nesterovs_momentum=True, power_t=0.5, random_state=None, shuffle=True, solver='lbfgs', tol=0.0001, validation_fraction=0.1.

Performance in average: 0.54 +/- 0.04

Table 13

Descriptions of the Parameters of the Multi-Layer Perceptron

Parameter	Description
hidden_layer_sizes	The ith element represents the number of neurons in the ith hidden layer.

Table 13

(Continued)

activation	Activation function for the hidden layer: 'identity', no-op activation, useful to implement linear bottleneck, returns f(x) = x 'logistic', the logistic sigmoid function, returns f(x) = 1 / (1 + exp(-x)). 'tanh', the hyperbolic tan function, returns f(x) = tanh(x). 'relu', the rectified linear unit function, returns f(x) = max(0, x)
solver	The solver for weight optimization: 'lbfgs' is an optimizer in the family of quasi-Newton methods. 'sgd' refers to stochastic gradient descent. 'adam' refers to a stochastic gradient-based optimizer
alpha	L2 penalty (regularization term) parameter.
batch_size	Size of mini-batches for stochastic optimizers. If the solver is 'lbfgs', the classifier will not use minibatch. When set to "auto", batch_size=min (200, n_samples)

Table 13

(Continued)

learning_rate	‘constant’ is a constant learning rate given by ‘learning_rate_init’. ‘invscaling’ gradually decreases the learning rate learning_rate_ at each time step ‘t’ using an inverse scaling exponent of ‘power_t’. effective_learning_rate = learning_rate_init / pow (t, power_t) ‘adaptive’ keeps the learning rate constant to ‘learning_rate_init’ as long as training loss keeps decreasing. Each time two consecutive epochs fail to decrease training loss by at least tol, or fail to increase validation score by at least tol if ‘early_stopping’ is on, the current learning rate is divided by 5.
learning_rate_init	The initial learning rate used. It controls the step-size in updating the weights. Only used when solver=’sgd’ or ‘adam’.
power_t	The exponent for inverse scaling learning rate. It is used in updating effective learning rate when the learning_rate is set to ‘invscaling’. Only used when solver=’sgd’.
max_iter	Maximum number of iterations. The solver iterates until convergence (determined by ‘tol’) or this number of iterations. For stochastic solvers (‘sgd’, ‘adam’), note that this determines the number of epochs (how many times each data point will be used), not the number of gradient steps.

Table 13

(Continued)

shuffle	Whether to shuffle samples in each iteration. Only used when solver='sgd' or 'adam'.
random_state	If int, random_state is the seed used by the random number generator; If RandomState instance, random_state is the random number generator; If None, the random number generator is the RandomState instance used by np.random.
tol	Tolerance for the optimization. When the loss or score is not improving by at least tol for two consecutive iterations, unless learning_rate is set to 'adaptive', convergence is considered to be reached and training stops.
momentum	Momentum for gradient descent update. Should be between 0 and 1. Only used when solver='sgd'.
nesterovs_momentum	Whether to use Nesterov's momentum. Only used when solver='sgd' and momentum > 0.
early_stopping	Whether to use early stopping to terminate training when validation score is not improving. If set to true, it will automatically set aside 10% of training data as validation and terminate training when validation score is not improving by at least tol for two consecutive epochs. Only effective when solver='sgd' or 'adam'

Table 13

(Continued)

validation_fraction	The proportion of training data to set aside as validation set for early stopping. Must be between 0 and 1. Only used if early_stopping is True
beta_1	Exponential decay rate for estimates of first moment vector in adam, should be in [0, 1]. Only used when solver='adam'
beta_2	Exponential decay rate for estimates of second moment vector in adam, should be in [0, 1]. Only used when solver='adam' epsilon: float, optional, default 1e-8 Value for numerical stability in adam. Only used when solver='adam'

4.2 Ensemble Methods

The Ensemble methods were also considered for this experiment. The goal of the Ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability/robustness over a single estimator [5].

There are two families of Ensemble methods in scikit-learn library- averaging methods and boosting methods [5]. In averaging methods, the driving principle is to build several estimators independently and then to average their predictions [5]. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced. Examples of this would be bagging methods and forests of randomized trees.

By contrast, in boosting methods, base estimators are built sequentially and one tries to reduce the bias of the combined estimator [5]. The motivation is to combine several weak models to produce a powerful ensemble. Examples of this would be AdaBoost and Gradient Tree Boosting.

4.2.A Ensemble Voting Classifier

The first ensemble method used for this research is the Ensemble Voting Classifier, which fits clones of the original estimators [5]. The main Of All the predictive classifiers and Bayes methods are combined and then applied to the Voting Classifier in order to improve generalizability/robustness over the ensemble estimator [5].

Table 14

Ensemble Voting Classifier

Model	Performance in average
Logistic Regression	0.54 +/- 0.03
K Nearest Neighbors	0.56 +/- 0.02
Support Vectors Machine	0.53 +/- 0.03
Random Forest	0.57 +/- 0.02
Multi-Layer Perceptron	0.54 +/- 0.04
Bernoulli Naive Bayes	0.56 +/- 0.03
Naive Bayes	0.55 +/- 0.04

Table 15

Descriptions for the Parameters of the Ensemble Voting Classifier

Parameter	Description
estimators	Invoking the fit method on the VotingClassifier will fit clones of those original estimators that will be stored in the class attribute self.estimators_. An estimator can be set to None using set_params.
voting	If 'hard', uses predicted class labels for majority rule voting. Else if 'soft', predicts the class label based on the argmax of the sums of the predicted probabilities, which is recommended for an ensemble of well-calibrated classifiers.
weights	Sequence of weights (float or int) to weight the occurrences of predicted class labels (hard voting) or class probabilities before averaging (soft voting). Uses uniform weights if None.
flatten_transform	Affects shape of transform output only when voting='soft' If voting='soft' and flatten_transform=True, transform method returns matrix with shape (n_samples, n_classifiers * n_classes). If flatten_transform=False, it returns (n_classifiers, n_samples, n_classes).

4.2.B Naive Bayes Methods

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features [5]. The different naive Bayes classifiers differ mainly by the assumptions they make in regarding to the

distribution of P (x_i \mid y). In spite of their apparently over-simplified assumptions, Naive Bayes classifiers have worked quite well in many real-world situations. They are famous for document classification and spam filtering. They require a small amount of training data to estimate the necessary parameters [5]. Theoretical reasons on why they work well is because the Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. The decoupling of the class conditional feature distributions meaning that their distribution can be independently estimated as a one-dimensional distribution.

4.2.C Bernoulli Naive Bayes

Like MultinomialNB, this classifier is suitable for discrete data. The difference is that while MultinomialNB works with occurrence counts, BernoulliNB is designed for binary/boolean features. In addition, this model is popular for document classification tasks where binary term occurrence features are used rather than term frequencies [5].

4.2.D Ensemble Adaptive Boosting Classifier

The second Ensemble method used for this research is Adaptive Boosting Classifier, which is a meta-estimator that begins by fitting a classifier on the original dataset [5]. It was first created by Yoav Freund and Robert Schapire [5]. It then fits additional copies of the classifier on the same dataset where the weights of incorrectly classified instances are adjusted such that the subsequent classifiers focus more on difficult cases. This classifier can be used together with other types of learning algorithms to improve their performance.

Best parameters of AdaBoostClassifier for this data set: algorithm='SAMME.R', base_estimator= DecisionTreeClassifier, learning_rate=1.0, n_estimators=50

Performance in average: 0.57 +/- 0.03

Table 16

Descriptions for the parameters of the Adaptive Boosting Classifier

Parameter	Description
base_estimator	The base estimator from which the boosted ensemble is built. Support for sample weighting is required, as well as proper classes_ and n_classes_ attributes.
n_estimators	The maximum number of estimators at which boosting is terminated. In case of perfect fit, the learning procedure is stopped early.
learning_rate	Learning rate shrinks the contribution of each classifier by learning_rate. There is a trade-off between learning_rate and n_estimators.
algorithm	If 'SAMME.R' then use the SAMME.R real boosting algorithm. base_estimator must support calculation of class probabilities. If 'SAMME' then use the SAMME discrete boosting algorithm. The SAMME.R algorithm typically converges faster than SAMME, achieving a lower test error with fewer boosting iterations.

4.2.E Ensemble Gradient Boosting Classifier

The third Ensemble method used for this research is Gradient Boosting Classifier, which builds an additive model in a forward stage-wise fashion. This allows for the optimization of arbitrary differentiable loss functions [5]. In each stage n_classes_ regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

Best parameters of Gradient Boosting Classifier for this data set: criterion='friedman_mse', init=None, learning_rate=0.1, loss='deviance', max_depth=3, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=100, presort='auto', random_state=None, subsample=1.0

Performance in average: 0.63 +/- 0.02

Table 17

Descriptions for the Parameters of the Gradient Boosting Classifier

Parameter	Description
loss	loss function to be optimized. 'deviance' refers to deviance (= logistic regression) for classification with probabilistic outputs. For loss 'exponential' gradient boosting recovers the AdaBoost algorithm.
learning_rate	learning rate shrinks the contribution of each tree by learning_rate. There is a trade-off between learning_rate and n_estimators.
n_estimators	The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance.
max_depth	Maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Tune this parameter for best performance; the best value depends on the interaction of the input variables.

Table 17

(Continued)

criterion	The function to measure the quality of a split. Supported criteria are “friedman_mse” for the mean squared error with improvement score by Friedman, “mse” for mean squared error, and “mae” for the mean absolute error. The default value of “friedman_mse” is generally the best as it can provide a better approximation in some cases.
min_samples_split	The minimum number of samples required to split an internal node: If int, then consider min_samples_split as the minimum number. If float, then min_samples_split is a percentage and ceil (min_samples_split * n_samples) are the minimum number of samples for each split.
min_samples_leaf	The minimum number of samples required to be at a leaf node: If int, then consider min_samples_leaf as the minimum number. If float, then min_samples_leaf is a percentage and ceil (min_samples_leaf * n_samples) are the minimum number of samples for each node.
min_weight_fraction_leaf	The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node. Samples have equal weight when sample_weight is not provided.

Table 17

(Continued)

subsample	The fraction of samples to be used for fitting the individual base learners. If smaller than 1.0 this results in Stochastic Gradient Boosting. subsample interacts with the parameter n_estimators. Choosing subsample < 1.0 leads to a reduction of variance and an increase in bias. The fraction of samples to be used for fitting the individual base learners. If smaller than 1.0 this results in Stochastic Gradient Boosting. subsample interacts with the parameter n_estimators. Choosing subsample < 1.0 leads to a reduction of variance and an increase in bias.
max_features	The number of features to consider when looking for the best split: If int, then consider max_features features at each split. If float, then max_features is a percentage and int(max_features * n_features) features are considered at each split. If “auto”, then max_features=sqrt(n_features). If “sqrt”, then max_features=sqrt(n_features). If “log2”, then max_features=log2(n_features). If None, then max_features=n_features. Choosing max_features < n_features leads to a reduction of variance and an increase in bias.

Table 17

(Continued)

max_leaf_nodes	Grow trees with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity. If None then unlimited number of leaf nodes.
min_impurity_split	Threshold for early stopping in tree growth. A node will split if its impurity is above the threshold, otherwise it is a leaf.
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value. The weighted impurity decrease equation is the following: N_t / N * (impurity - N_t_R / N_t * right_impurity - N_t_L / N_t * left_impurity) where N is the total number of samples, N_t is the number of samples at the current node, N_t_L is the number of samples in the left child, and N_t_R is the number of samples in the right child. N, N_t, N_t_R and N_t_L all refer to the weighted sum, if sample_weight is passed.
init	An estimator object that is used to compute the initial predictions. init has to provide fit and predict. If None it uses loss.init_estimator.

Table 17

(Continued)

presort	Whether to presort the data to speed up the finding of best splits in fitting. Auto mode by default will use presorting on dense data and default to normal sorting on sparse data. Setting presort to true on sparse data will raise an error.

4.2.F Ensemble Bagging Classifier

The fourth Ensemble method used is the Ensemble Bagging classifier, which is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregates their individual predictions (either by voting or by averaging) to form a final prediction [5]. A meta-estimator can typically be used as a means to reduce the variance of a black-box estimator (e.g., a decision tree) by introducing randomization into its construction procedure and then making an ensemble out of it.

This algorithm encompasses several works from literature. When random subsets of the dataset are drawn as random subsets of the samples, it is known as Pasting [5]. If samples are drawn with replacement, then the method is known as Bagging [5]. When random subsets of the dataset are drawn as random subsets of the features, it is known as Random Subspaces [5]. Finally, when base estimators are built on subsets of samples and features, then the method is known as Random Patches [5].

Best parameters of Bagging Classifier for this data set: class_weight=None, criterion='gini', max_depth=None, max_features=None, max_leaf_nodes=None, min_impurity_decrease=0.0,

min_impurity_split=None, min_samples_leaf=1, min_samples_split=2,

min_weight_fraction_leaf=0.0, presort=False, random_state=None, splitter='best'.

Performance in average: 0.58 +/- 0.03

Table 18

Descriptions for the Parameters of the Ensemble Bagging Classifier

Parameter	Description
base_estimator	The base estimator to fit on random subsets of the dataset. If None, then the base estimator is a decision tree.
n_estimators	The number of base estimators in the ensemble.
max_samples	The number of samples to draw from X to train each base estimator. If int, then draw max_samples samples. If float, then draw max_samples * X.shape[0] samples. max_features: int or float, optional (default=1.0) The number of features to draw from X to train each base estimator. If int, then draw max_features features. If float, then draw max_features * X.shape[1] features.
bootstrap	Whether samples are drawn with replacement.
bootstrap_features	Whether features are drawn with replacement.
oob_score	Whether to use out-of-bag samples to estimate the generalization error.

4.3 Summary of Classification Results

The experimental data shows that Ensemble methods worked better in helping to derive accurate predictions. Part of the reasoning behind the poor prediction is due to not being able to calculate the time-lag between the release date of the news and the date the DJI prices adjusted accordingly. Not all the prices can be shifted to the next days since there are already many missing days in the time series (holidays and weekends). Table 6 shows the summarized average accuracy per model.

Table 19

Average Performance of the Classifiers

Machine Learning Model	Performance in average
Logistic Regression Classifier	0.54 +/- 0.03
K Nearest Neighbors	0.56 +/- 0.02
Support Vectors Machine	0.53 +/- 0.03
Random Forest Classifier	0.57 +/- 0.02
Multi-Layer Perceptron Classifier	0.54 +/- 0.04
Bernoulli Naive Bayes (Ensemble)	0.56 +/- 0.03
Naive Bayes (Ensemble)	0.55 +/- 0.04
Ensemble Adaptive Boosting Classifier	0.57 +/- 0.03
Ensemble Gradient Boosting Classifier	0.63 +/- 0.02
Ensemble Bagging Classifier	0.58 +/- 0.03

4.4 Prediction

After determining that there is a meaningful link between the DJI related News Sentiments and the DJI price movements, we can predict the movements of DJI prices. Since we cannot predict the future prices using the news of 31 months, the data set needs to be broken into smaller sets. To do so, a short period yet not too short that the machine learning models cannot learn the patterns, is most beneficial. To learn the patterns in the smaller data sets, there should be enough samples of two desired outputs (up or down) in each data set. The number of samples in each data set should not be less than 20 since there is at least one set that does not have at least one sample of each the desired outputs.

The mean score and the 95% confidence interval of different predictive machine learning models using a different number of samples (in the range of 20 and 50 days) have been calculated and the results are shown in table 20.

Table 20

Prediction Results Based on the Number of Samples in each Data Set

Model	20-sample	25-sample	30-sample	35-sample
Average Prediction Performance	0.58 +/- 0.03	0.62 +/- 0.04	0.67 +/- 0.02	0.65 +/- 0.03

These results show that there is a tradeoff in choosing the size of sample sets. In General, the more training samples in each set result the better classification performance. On the other hand, we cannot relate the old news to the price movements. Having less or more than 30 samples in each sample set results in less prediction accuracy in average. Due to this, the time series was broken into 30-sample sets to predict the unseen prices of DJI. This break was done by shifting

data to the next day. The input of the first sample set includes the News Sentiments of from 1st to 30th day, and the output includes the DJI price movements of 3rd to 33rd day. The input of the second sample set includes the News Sentiments of from 2nd to 31st day, and the output includes the DJI price movements of days from 4th to 34th day. By doing so, 615 sample sets were created, the input of the last sample set includes the News Sentiments of 615th to 645th day, and the output includes the DJI price movements of days from 617th to 647th day.

To train the predictive models, the News Sentiments and the desired outputs of 28 days of each sample set were used to train the predictive models. Having trained the models, the News Sentiments of the last 2 days of the sample set were applied to the trained models to predict, and the desired outputs of the last 2 days (test set) were compared to the predicted outputs to test the performance of the predictions. Since the outputs were shifted to 2 next days we can consider the outputs of last 2 days of each sample set as future and unseen price movements.

The mean score and the 95% confidence interval of each classification model are calculated, and the results are shown in Table 21.

Table 21

Average Prediction Performance

Model	Average Accuracy
Average Prediction Performance	0.67 +/- 0.02

Based on these results, we can claim that having the DJI related News Sentiments of recent 30 days we can predict the DJI price movements of 2 days in the future with an accuracy of 0.67.

CHAPTER 5

CONCLUSION AND FUTURE PERSPECTIVES

This research employed GDELT dataset, multiple classification models—Logistic Regression Classifier, K Nearest Neighbors, Support Vectors Machine, Random Forest Classifier, Multi-Layer Perceptron Classifier—to find the best correlation possible in this data between the News Sentiments and the movements of the DJI prices. Having found the best possible correlation between the News Sentiments and the movements of the DJI prices, the data set was broken into smaller sets so that movements of unseen prices can be predicted. Each data set was broken into train sets and test sets, and the predictive models were trained using the train sets. The movements of DJI unseen prices of 2 days in the future (test set) were predicted by the trained predictive models using the News Sentiments of the sample set. The performances of the predictions were measured by comparing the predicted price movements and the real price movements.

In addition, Grid Search Cross-Validation and Ensemble methods were utilized to improve prediction performances in forecasting the daily movement direction of one of the most popular New York Stock Exchange indices.

Based on the experiment results of this research concludes that there is a significant link between relevant financial news and the movements of DJI prices to predict the movements of the unseen prices. In addition, on contrary to the famous financial hypothesis that states the Stock Market is not predictable, it can be concluded that the Stock Market movement is predictable. We can claim that the more accurately we analyze the financial news the more accurately we can predict the unseen prices. This study not only shows that there is a significant

link between DJI related news and the DJI prices, but also shows how to use the news to predict the future prices. However, this research did have gaps that should be considered for future studies as it may in fact help raise the accuracy of predictability.

One gap was in finding time lag. Finding the time lag for each day might help with the accuracy especially because in this study the time lag was not consistent due to holidays and weekends. Help from some financial experts would help with determining the best way to find time lag. In addition, future works can also consider using Amazon Mechanical Turk to find the time lags for each. Another way to tackle this problem may be by using interpolation to assign some prices for the missing days to complete the time series.

Future studies can look into the Deep Learning techniques to predict the future/unseen prices. This method was not mentioned in this research because they were unsuccessful to use due to the heavy computations that needed to be distributed over several powerful systems. Deep Learning techniques would be a great future perspective for this research. In addition, another direction for future studies would include finding some thresholds for classifying price movements in order to increase the accuracy of prediction.

REFERENCES

[1] All GDELT Event Files, data.gdeltproject.org/events/index.html

[2] Bing, L., Chan, K., OU, C. (2014). Public Sentiment Analysis in Twitter Data for Prediction of a Company's Stock Price Movements. IEEE 11th International Conference on e-Business Engineering.

[3] Birgul A. (2003). Stock Market Prediction Using Artificial Neural Networks.

[4] Cao, L. (2002). Support Vector Machines Experts For Time Series Forecasting.

[5] Dietterich, T. Ensemble Metho Ds in Machine Learning. Oregon State University, 2011, web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf.

[6] Fallahi, F. Machine Learning on Big Data For Stock Market Prediction. Aug. 2017.

[7] Gruhl, D., Guha, R., Kumar, R., Novak, J., Tomkins, A. (2005). The Predictive Power of Online Chatter In Proceedings of the Eleventh International Conference on Knowledge Discovery in Data Mining (pp. 78-87). ACM.

[8] Jiao, P., Veiga, A., Walther, A. (2016). Social Media, News Media and the Stock Market.

[9] Johan B., Mao, H., and Zeng, X. (2011), Twitter mood predicts the stock market, Journal of Computational Science, 2(1), March 2011, Pages 1-8.

[10] Joshi, K., Bharathi H., Rao, J. Stock Trend Prediction Using News Sentiment Analysis.

[11] Khaidem, L., Saha, S., Dey, S. (2016). Predicting the Direction of Stock Market Prices Using Random Forest.

[12] Killer, M. “Textblob Documentation.” 17 Sept. 2017, pp. 1–45. https://media.readthedocs.org/pdf/textblob-de/latest/textblob-de.pdf.

[13] Kim, K. (2003). Financial Time Series Forecasting Using Support Vector Machines.

[14] Kimmey, D., and Yoo, J. (2016). Nowcasting With Social Media Data.

[15] Laney, D. (2001). 3-D Data Management: Controlling Data Volume, Velocity and Variety, Technical report, META Group.

[16] Lee, J. (2001). Stock Price Prediction Using Reinforcement Learning.

[17] Madge, S. (2015). Predicting Stock Price Direction using Support Vector Machines.

[18] Mittal, A., and Goel, A.. “Stock Prediction Using Twitter Sentiment Analysis.” Standford University

[19] Moat, H. (2013). Quantifying Wikipedia Usage Patterns Before Stock Market Moves.

[20] Moody, J., Wu, L., Liao, Y., Saffell, M. (1998). Performance Functions And Reinforcement Learning For Trading Systems And Portfolios.

[21] Nieminen, P. (February 20th, 2012). Classification and Multilayer Perceptron Neural Networks. users.jyu.fi/~nieminen/dm2012mlp/dm_mlp.pdf.

[22] Nikola Milosevic Equity Forecast: Predicting long term stock price movement using machine learning

[23] Porshnev, A., Lakshina, V., Redkin, I. (2016). Could Emotional Markers in Twitter Posts Add Information to the Stock Market.

[24] Qiao, F., and X. Zhang. “Predicting Social Unrest Events with Hidden Markov Models Using GDELT.” Discrete Dynamics in Nature and Society, 10 May 2017, www.hindawi.com/journals/ddns/2017/8180272/.

[25] Schrodt, P. (2012). CAMEO Conflict and Mediation Event Observations Event and Actor Codebook.

[26] SciKits, https://www.scipy.org/scikits.html.

[27] Scikit Website. http://scikit-learn.org/stable/.

[28] Shah V. (2007). Machine Learning Techniques For Stock Prediction.

[29] Shalev-Shwartz, S. (2014). “Understanding Machine Learning: From Theory to Algorithms.” http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf.

[30] Shen, S., Jiang, H., Zhang, T. (2012). Stock Market Forecasting Using Machine Learning Algorithms.

[31] Tianxin Dai, Arpan Shah, Hongxia Zhong (2012). Automated Stock Trading Using Machine Learning Algorithms .

[32] Wei F., Bifet, A. (2012). Mining Big Data: Current Status, and Forecast to the Future, ACM SIGKDD Explorations Newsletter table of contents archive, Vol. 14 Issue 2.

[33] Yu, Y., Duan, W., Cao, Q. (2013). The Impact of Social and Conventional Media on Firm Equity Value: A Sentiment Analysis Approach.

[34] Yuan, Y. (2016). Modeling Inter-Country Connection from Geotagged News Reports: A Time-Series Analysis.

[35] Zhou, Z. (November 4th,2014). Data Opportunities and Challenges: Discussions from Data Analytics Perspectives, IEEE Computational Intelligence Magazine Vol: 9, Issue: 4.

[36] Robots have been running the US stock market, and the government is finally taking control qz.com/370019/

[37] (2012). 84% of All Stock Trades Are By High-Frequency Computers www.zerohedge.com/contributed/2012-17-26/84-all-stock-trades-are-high-frequency-computers [38] GDELT on Google BigQuery, bigquery.cloud.google.com/table/gdelt-bq:full.events

APPENDICES

APPENDIX A

Part of the SQL Query Used to Retrieve the Data Set

```
SELECT sub2.EVENTDATE      AS EVENT_DATE,
COUNT(sub2.SOURCEURL)    AS COUNT_SOURCEURL,
AVG(sub2.GOLDSTEINSCALE) AS Avg_GOLDSTEINSCALE,
AVG(sub2.NUMMENTIONS)    AS Avg_NUMMENTIONS,
AVG(sub2.NUMSOURCES)     AS AVG_NUMSOURCES,
AVG(sub2.NUMARTICLES)    AS AVG_NUMARTICLES,
AVG(sub2.AVGTONE)        AS AVG_AVGTONE
FROM
(SELECT sub1.*
FROM
(SELECT sub0.*
FROM
(SELECT *
FROM GDELT
WHERE NUMMENTIONS >= 14
AND NUMSOURCES    >= 3
AND NUMARTICLES   >= 14
AND ISROOTEVENT   > 0
) sub0
WHERE COUNTRYCODE LIKE '%US%'
```

```
OR COUNTRYCODE LIKE '%UK%'
OR COUNTRYCODE LIKE '%JPN%'
OR COUNTRYCODE LIKE '%DEU%'
OR COUNTRYCODE LIKE '%CHN%'
)sub1
WHERE SOURCEURL LIKE '%Dow-Jones%'
OR SOURCEURL LIKE '%DJIA%'
OR SOURCEURL LIKE '%Industrial-Average%'
OR SOURCEURL LIKE '%Jones-Industrial%'
OR SOURCEURL LIKE '%^DJI%'
OR SOURCEURL LIKE '%Dow-30%'
) sub2
GROUP BY (sub2.EVENTDATE)
ORDER BY (sub2.EVENTDATE);
```

APPENDIX B

An Example of the Python Codes used for the Predictions

```
data = pd.read_csv('C:/Users/Majid/Documents/Database/Export4_Lag5.csv')

X = data[['AVG_GOLDSTEINSCALE','AVG_AVGTONE']]

y = data['Price_Change'].as_matrix().astype(int)

scores = []

n = 30

for i in range(0,646-n,1):

        Xi = X[i:(i+n)]

        yi = y[i:(i+n)]

        scaler = StandardScaler()

        Xi = scaler.fit_transform(Xi)

        Xi_train,Xi_test,yi_train,yi_test =
    train_test_split(Xi,yi,test_size=0.1,random_state=None,shuffle=False)

        clf = RandomForestClassifier()

        clf.fit(Xi_train,yi_train)

        yi_pred = clf.predict(Xi_test)

        score = metrics.accuracy_score(yi_test,yi_pred)

        scores.append(score)
```

```
        print('Subset #',i,' ( from',i,'to',i+30,')',)
        print('AVG_GOLDSTEINSCALE AVG_AVGTONE\n',Xi)
        print('\n')
        print('Price Movements:',yi)
        print('\n')
        print('Score:',score)
print('Average Score:',sum(scores)/len(scores))
```

APPENDIX C

An Example of Python Codes used for the Classifications

```
data = pd.read_csv('~/5-DAY-LAG.csv')

X = data[['AVG_GOLDSTEINSCALE','AVG_AVGTONE']]

y = data['Price_Change'].as_matrix().astype(int)

scaler = StandardScaler()

X = scaler.fit_transform(X)

clf1 = LogisticRegression(C=1, class_weight=None, dual=False, fit_intercept=True,
        intercept_scaling=1, max_iter=200, multi_class='ovr', n_jobs=1,
        penalty='l2', random_state=None, solver='newton-cg', tol=0.0001,
        verbose=0, warm_start=False)

clf2 = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
         max_depth=None, max_features='auto', max_leaf_nodes=None,
         min_impurity_decrease=0.0, min_impurity_split=None,
         min_samples_leaf=1, min_samples_split=2,
         min_weight_fraction_leaf=0.0, n_estimators=7, n_jobs=1,
         oob_score=False, random_state=None, verbose=0,
         warm_start=False)
```

APPENDIX C

(Continued)

```
clf3 = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
        metric_params=None, n_jobs=1, n_neighbors=17, p=2,
        weights='uniform')
clf4 = svm.SVC(C=1, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
clf5 = MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
      beta_2=0.999, early_stopping=False, epsilon=1e-08,
      hidden_layer_sizes=(100,), learning_rate='invscaling',
      learning_rate_init=0.001, max_iter=200, momentum=0.9,
      nesterovs_momentum=True, power_t=0.5, random_state=None,
      shuffle=True, solver='lbfgs', tol=0.0001, validation_fraction=0.1,
      verbose=False, warm_start=False)
clf6 = BernoulliNB()
eclf = VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('knn',
clf3),('svm',clf4),('mlp',clf5),('gnb',clf6)], voting='hard')
```

APPENDIX C

(Continued)

```
for clf, label in zip([clf1, clf2, clf3, clf4, clf5, clf6, eclf], ['Logistic Regression','Random Forest',
'KNeighborsClassifier','SVM','MLPClassifier','BernoulliNB','Ensemble']):
    scores = cross_val_score(clf, X, y, cv=5, scoring='accuracy')
    print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))
```

VITA

Graduate School

Southern Illinois University

Majid Memari

memari.majid@hotmail.com

Azad Tehran University

Bachelor of Industrial Engineering May 2010

Azad Qazvin University

Master of Business Administration (MBA), May 2015

Thesis Title:

Predicting the Stock Market Using News Sentiment Analysis

Major Professor: Dr. Norman Carver