

---

[All ETDs from UAB](#)

[UAB Theses & Dissertations](#)

---

2017

## A Framework For Social Network Sentiment Analysis Using Big Data Analytics

Bharat Sri Harsha Karpurapu  
*University of Alabama at Birmingham*

Follow this and additional works at: <https://digitalcommons.library.uab.edu/etd-collection>

---

### Recommended Citation

Karpurapu, Bharat Sri Harsha, "A Framework For Social Network Sentiment Analysis Using Big Data Analytics" (2017). *All ETDs from UAB*. 2108.  
<https://digitalcommons.library.uab.edu/etd-collection/2108>

This content has been accepted for inclusion by an authorized administrator of the UAB Digital Commons, and is provided as a free open access item. All inquiries regarding this item or the UAB Digital Commons should be directed to the [UAB Libraries Office of Scholarly Communication](#).

A FRAMEWORK FOR SOCIAL NETWORK SENTIMENT ANALYSIS USING  
BIG DATA ANALYTICS

by

BHARAT SRI HARSHA KARPURAPU

LEON JOLOLIAN, COMMITTEE CHAIR  
KARTHIKEYAN LINGASUBRAMANIYAN,  
MURAT M. TANIK,

A THESIS

Submitted to the graduate faculty of The University of Alabama at Birmingham  
in partial fulfillment of the requirements for the degree of  
Master of Science

BIRMINGHAM, ALABAMA

2017

Copyright © by  
Bharat Sri Harsha Karpurapu  
2017

# **A FRAMEWORK FOR SOCIAL NETWORK SENTIMENT ANALYSIS USING BIG DATA ANALYTICS**

**BHARAT SRI HARSHA KARPURAPU**

**MASTER OF SCIENCE IN ELECTRICAL ENGINEERING**

## **ABSTRACT**

The primary research of this thesis focused on the development of a Big Data framework for performing sentiment analysis on social networking sites. Over the last decade, social media has been gaining lots of popularity for sharing thoughts and feelings with a user base of over two billion users. Social networking sites such as Twitter, Facebook, and Instagram are increasingly becoming huge repositories of thoughts and opinions on a wide variety of topics. Several public and private organizations, such as Government and companies, are attempting to exploit the expressed preferences, opinions, and attitudes regarding politics, commercial products and other matters of personal importance towards a competitive edge. One of the efficient ways to get this information is by performing sentiment analysis on these electronic repositories. With the data being ubiquitous, the bottlenecks here are processing speed, storage, and time involved in the traditional storage system. Therefore, in order to deal with the data processing of these massive amounts of data, some special tools and techniques have been offered by Big Data framework. Much of the current work involving sentiment analysis is performed on online customer reviews, blogs, forums, etc., using lexical and machine learning methods by adopting batch processing. In this thesis, we offer an innovative approach for developing a generalized Big Data framework for performing social network sentiment analysis. This framework consists of a real-time collection of streamed data followed by machine learning methods for performing sentiment analysis.

For getting data and for performing sentiment analysis, a live mini-blogging website ‘Twitter’ is considered which generates tweets at the rate of 6000 per second with an inherent restriction of no more than 140 characters per tweet. For processing speeds and storage capacity, a large-scale cluster-computing framework like the Apache Spark was used, due to its capacity in handling data and at high speed. As for the sentiment analysis on the collected tweets, the Naive Bayes algorithm was used. The final generalized framework was tested on two case studies using different problem scenarios with considerable success.

Keywords: Social Media, Big Data Analytics, Sentiment Analysis, Apache Spark, Naive Bayes.

## DEDICATION

I would like to dedicate this research to my loving family and good wishers. A precious feeling of gratitude to my parents, Hanuman Babu K. and Vardhani K., who showered their love and support for all through my life and making me for what I am today.

My special dedication is to my brother Rakesh K.,

*who sacrificed his happiness for mine,*

*who sacrificed his comfort for mine,*

*who sees his success in mine, and*

*who is my constant motivation and energy.*

This thesis is also dedicated to my sister in law, Surya Sreedhari K., for her continuous support, love and encouragement.

## **ACKNOWLEDGEMENTS**

Firstly, I would like to express my deep gratitude to my thesis chair, Dr. Leon Jololian, for his keen attention, patience, and constant encouragement. I am very thankful to him for giving me ample of time to explore my areas of interest, which is the burning requirement for any research.

I am also thankful for my committee members Dr. Karthikeyan Lingasubramanian and Dr. Murat M. Tanik, for their sheer support and guidelines during the whole research. I am blessed to have them as my committee members.

I would also like to acknowledge Dr. Thomas Anthony, Director of Big Data Research and Analytics Lab for empowering me with his ideas and methodologies. I am also grateful to Shyam Prabhakar, Data Science Manager of Regions Bank for his assistance and timely advice throughout my whole thesis.

Last not but not the least, I would like to extend my acknowledgements to my friends for their support, fun and encouragement, and to each and every member of UAB community.

## TABLE OF CONTENTS

	<i>Page</i>
ABSTRACT .....	iii
DEDICATION .....	v
ACKNOWLEDGMENTS .....	vi
LIST OF TABLES .....	x
LIST OF FIGURES .....	xi
 1. INTRODUCTION .....	 1
1.1 Motivation .....	1
1.2 Social Media and its Ubiquitous Nature .....	3
1.2.1 Twitter .....	4
1.3 Social Network Sentiment Analysis and its Role .....	4
1.4 Big Data Analytics .....	5
1.4.1 Spark and its Importance .....	6
1.5 Research Questions .....	6
1.6 Thesis Documentation Outline .....	7
 2. LITERATURE REVIEW AND THEORITICAL BACKGROUND .....	 9
2.1 Sentiment Analysis .....	9
2.1.1 Types of Sentiment Analysis .....	10
2.1.1.A Document Level Sentiment Analysis .....	10
2.1.1.B Sentence Level Sentiment Analysis .....	11
2.1.1.C Aspect Level Sentiment Analysis .....	12
2.1.2 Sentiment Analysis Process .....	12
2.1.2.A Direct Opinion .....	13
2.1.2.B Comparative Opinion .....	14
2.2 Machine Learning .....	14
2.2.1 Data Oriented Machine Learning .....	15
2.2.1.A Learning Process .....	16



2.2.2 Supervised Vs Unsupervised .....	18
2.2.2.A Regression Vs Classification.....	18
2.2.3 Parametric Vs Non Parametric.....	18
2.2.4 Naïve Bayes Algorithm.....	19
2.2.5 Metrics of Machine Learning.....	22
2.3 Big Data .....	24
2.3.1 Attributes of Big Data.....	25
2.3.1.A Volume .....	26
2.2.1.B Velocity.....	26
2.2.1.C Variety .....	27
2.2.1.D Veracity .....	27
2.3.2 Hadoop.....	27
2.3.2.A Hadoop Architecture .....	28
2.4 Summary .....	30
 3. TOOLS BACKGROUND .....	 31
3.1 Profile of Tools Architecture .....	31
3.2 Anaconda Framework.....	32
3.3.1 Anaconda Navigator .....	32
3.3.2 Conda .....	33
3.3 Jupyter Notebook.....	35
3.4 Spark .....	37
3.4.1 Data Abstractions in Spark .....	37
3.4.2 Architecture of Spark.....	38
3.4.3 Spark Workflow in Proposed Framework .....	40
3.5 Summary .....	41
 4. SYSTEM ARCHITECTURE AND IMPLEMENTATION.....	 42
4.1 Top Level Architecture of Proposed Framework .....	42
4.2 Process Flow .....	44
4.2.1 Data Collection .....	44
4.2.2 Preprocessing.....	46
4.2.3. Data Modelling.....	47
4.3 Case Study .....	48
4.3.1 Case Study 1 .....	48
4.3.2 Case Study 2 .....	49
4.4 Summary .....	49
 5. RESULTS AND ANALYSIS.....	 50
5.1 Case Study1 .....	50

5.1.1 Exploratory Analysis .....	50
5.1.2 Summary of Key Findings .....	55
5.1.3 Data Modelling .....	56
5.1.4 Final Analysis .....	58
5.2 Case Study 2 .....	59
5.3 Summary .....	60
 6. CONCLUSION AND FUTURE SCOPE .....	 61
6.1 Conclusion .....	61
6.2 Future Scope .....	62
 LIST OF REFERENCES .....	 63
 APPENDIX: PROGRAM CODE .....	 66

## LIST OF TABLES

<i>Table</i>	<i>Page</i>
2.1 Classification of Learning algorithms.....	19
2.2 Confusion Matrix .....	22

## LIST OF FIGURES

<i>Figure</i>	<i>Page</i>
1.1 Users Distribution in Social Media.....	3
2.1 Evolution of Machine Learning.....	15
2.2 Machine Learning Process.....	17
2.3 Attributes of Big Data.....	25
2.4 Hadoop Architecture.....	28
3.1 Profile of Tools Architecture .....	31
3.2 Anaconda Navigator .....	32
3.3 Command Interface of Conda.....	33
3.4 A View of Jupyter Notebook by highlighting different ways to write .....	35
3.5 Notebook by highlighting different downloadable options .....	36
3.6 Sample of a complete Notebook.....	36
3.7 Architecture of Spark.....	38
3.8 Spark work flow in Proposed Framework .....	40
3.9 sc and sqlContext variables in Jupyter Notebook.....	41
4.1 Top Level Architecture of the Framework .....	43
4.2 Process Flow in Proposed Framework.....	44
4.3 Attributes of the returned tweet .....	45
4.4 Preprocessing .....	46

5.1	Language Distribution of Tweets.....	51
5.2	Country wide Distribution of Tweets with Count Variation .....	52
5.3	Demographic Distribution of Tweets.....	53
5.4	Dashboard View from Tableau.....	54
5.5	Company Distribution of Tweets.....	55
5.6	Polarity Distribution of Tweets.....	56
5.7	Sample Output of Case Study I.....	57
5.8	Sentiment Score of Different Automobile Companies .....	58
5.9	Sample Output of Case Study II .....	59
5.10	Action Email .....	59

**CHAPTER – 1**  
**INTRODUCTION**  
**1.1 MOTIVATION**

Traditionally, surveys were used as one of the major methods for finding out the opinion of a group of people about a particular topic. However, over the last two decades with the proliferation of Web to the social media sites such as Twitter, Facebook, and Tumblr, social media are increasingly becoming the platform of choice for people to express their views or opinions. With an account of over two billion users, social media offers a plethora of user-generated content, which can be read by any interested party. The exponential growth of users on the social media provides a major source for gathering people moods and opinions. This generated content from the social media can range anywhere from politics, consumer products, companies, entertainment, etc. The vast amount of data generated are systematically stored in repositories throughout various data centers around the world. This rich data is available for expert people such as researchers, data analysts, and data scientists, to explore and formulate, patterns and trends in public opinion on a large-scale.

Among all the channels in the social media, Messaging and Microblogging become an important means where people can express their views on various topics of their interests or share their life experiences and much more. Particularly Twitter, a microblogging website where people are restricted to voice their views within 140 words,

offers a good platform for researchers and analysts to extract the sentiment from these small microblog tweets. However, to analyze this huge amount of data and to extract the meaning from the text raises the need for automated processing. The use of Big Data methods makes it possible for this automated processing on these huge amounts of data. These innovations and requirements give the opportunity for researchers to develop a large-scale cluster computing framework spark, which performs real-time streaming and processing of data from social networking sites.

Current Sentiment analysis techniques are evolved from the field of natural language processing, computational linguistics, artificial intelligence, text analytics and machine learning. Sentiment Analysis of social media sites through machine learning methods involves a sample of data to train, and there are many research works published in this area. People performed the sentiment analysis with a various combination of tools and machine learning methods. However, there is no work which concentrated on developing a generalized Big Data framework for performing social network sentiment analysis which can apply to many numbers of use cases.

This thesis takes into consideration of various related works and aims at developing a real-time Big Data framework for social network sentiment analysis. It also attempts to demonstrate the uniqueness of this approach by performing case studies.

## 1.2 SOCIAL MEDIA AND ITS UBIQUITOUS NATURE

Kaplan and Haenlein [1] defined social media as the “group of internet applications that built on the ideological and technological foundations of Web 2.0, and that allows the creation and exchange of user-generated content.”

According to statistics of SEJ (Search Engine Journal), more than 2 billion people [2] (almost 35% of world’s population) are social media users. The major contributors are Facebook, Twitter, Instagram, LinkedIn, Google+, YouTube, etc. Facebook occupies the first position with 1.5 billion users, and Twitter is in the top 10 with 316 million users. With these great number of users in the social media, the data generated by these sites is also high. For instance, Facebook generates almost 4 million posts every day.

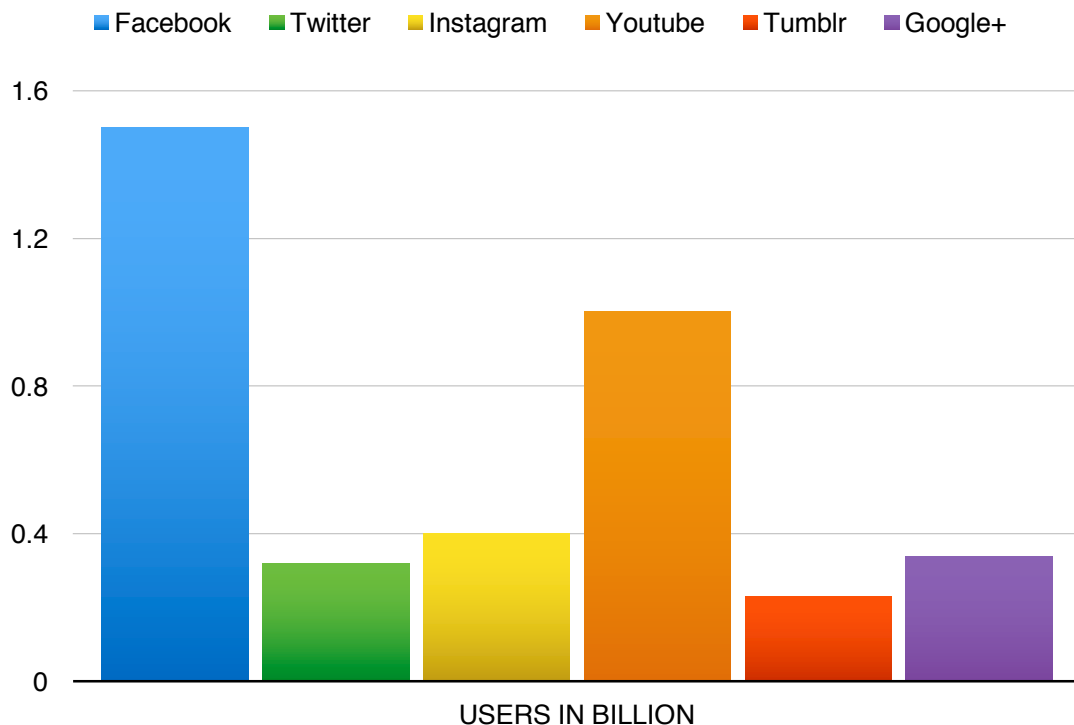


Figure 1.1: Users Distribution in Social Media [2]



People are adopting social media as a platform to express their opinions on various things such as movies, products, and politics. So social media offers a good source of information for gathering the people opinions and views.

### ***1.2.1 TWITTER:***

Twitter, a microblogging website offers an efficient platform for performing sentiment analysis. People express their opinion in the form of small messages called tweets, which can be displayed as publicly or can be restricted to a particular group.

Twitter has been selected for the following reasons:

- It is a common open-source platform.
- By limiting the maximum number to 140 words for a tweet, Twitter also forces the user to express his or her opinion in that limit.
- There is no need to pay for Twitter API, and it is very easy to mine them as real-time tweets, and it has an extensive collection of API's [3].
- Lastly, it is a Big Pool of sentiments.

## **1.3 SOCIAL NETWORK SENTIMENT ANALYSIS AND ITS ROLE**

Sentiment Analysis is an area evolved from the combination of natural language processing, text analytics, and computational linguistics. It is the process of determining the sentiment or opinion or mood of the particular document or text expressed by the author. It assigns a particular polarity to the text either positive, negative or neutral; this analysis helps in determining the state of the person such as happy, angry and sad.

Social network sentiment analysis involves the analysis performing on the data available from the social networking sites. Social networking sites are becoming a rich source of user-generated content with the number of users increasing daily. People express many views regarding a particular domain in different means such as blogs and posts. As it is becoming a huge platform of user-generated content, analyzing this content regarding the product or movie or the government is crucial in any domain for developing their plans.

#### **1.4 BIG DATA ANALYTICS**

Big Data is the data whose scale, complexity, diversity, and velocity require some new architecture, infrastructure, algorithms to manage and extract patterns from it. It is characterized by four features volume, velocity, veracity, and variety.

The field of Big Data offers various tools and technologies to work with this enormous amounts of data. The most well known tools include Hadoop, Spark, Hive, Pig, etc. This available number of tools makes it easier to extract and work with higher levels of data. There are tools available for real-time streaming such as Spark Streaming, Apache Flume, and Apache Kafka and for performing real-time data operations.

Social networking sites are becoming the valuable sources of these big data; Facebook is dealing with 500 TB of data every day. More than 100 hours of videos are being uploaded to YouTube even minute [36]. Companies such as Google, Yahoo and Bing are recording search results every day for analytics purposes. Big Data Analytics makes it easier to perform the data analytics on this huge amount of information.

#### ***1.4.1 SPARK AND ITS IMPORTANCE:***

Spark is a fast cluster-computing framework developed at the University of California at Berkley. The primary abstraction in Spark is RDD's (Resilient Distributed Datasets). Spark is highly dominant in the Big Data field due to the following features:

- It offers a unified programming abstraction model, which supports a variety of workload applications such as batch processing, streaming, and iterative processing.
- It is 100x faster than Map Reduce applications and comparably faster than any application of the specialized systems.
- It achieves efficiency and fault tolerance across various nodes at very low cost.

### **1.5 RESEARCH QUESTIONS**

**Question1: How would it be possible to parameterize domain characteristics in order to allow the framework to specialize for various domains?**

The whole idea of this thesis is based on this question, and the answer to this is “Yes.” The framework, which we developed here can be used for performing sentiment analysis on various domains such as Banking, Automobiles and Airlines by changing domain characteristic values. Here in this thesis, we performed sentiment analysis on Automobile Industry as a POC (Proof of Concept) and as case study I. This framework was extended at the end with an action email system, which we performed on Regions Bank as a case study II.

**Question2: How would it be possible to create an upgradable framework such that components can be replaced whenever new tools with better performance become available?**

Yes, we can build an upgradable framework. Here in the development of this framework, we used different tools such as Spark, Jupyter Notebook, Anaconda, Tableau, and Plotly. However, for developing the framework, we followed a layered architecture, which makes it possible to replace any layer or any part in the layer with a new efficient tool/technology in the future.

## **1.6 THESIS DOCUMENTATION OUTLINE**

The remaining chapters are organized as follows:

- Chapter – 2, Literature Review and Theoretical Background, this section focuses on providing the literature review and Theoretical background on topics like Sentiment Analysis, Big Data Analytics, and Machine Learning.
- Chapter – 3, Tools Background, this section provides information about the tools used in developing the proposed framework.
- Chapter – 4, System Architecture and Implementation, this section concentrates on giving details about the architecture the proposed framework and describes the implementation process flow in the framework.
- Chapter – 5, Results and Analysis, this section provides the Results of the case studies I and II with some analysis and evaluation.

- Chapter – 6, Conclusion and Future Scope, this is the final chapter in the thesis documentation which provides the conclusion and future scope of the developed framework.

## **CHAPTER – 2**

### **LITERATURE REVIEW AND THEORETICAL BACKGROUND**

This chapter focuses on providing Literature Review and the Theoretical Background for the proposed framework.

#### **2.1 SENTIMENT ANALYSIS**

Sentiment Analysis is one of the fields in the area of natural language processing which aims at determining the writer's attitudes, emotions or moods from the text regarding various topics. It tries to extract the underlying mood or emotion expressed by the author. It involves identifying the opinion holder and the entity, which is the target. The important task in sentiment analysis is to identify the object i.e. opinion or subjectivity. Quirk [4] defined subjectivity as “private state of something that is not open to objective observation or verification.” The word private state may indicate attitude, opinion, mood, emotion, etc. Based on the Quirks definition of subjectivity, Wiebe [5], the most prominent natural language processing expert, defined subjectivity as a tuple (p, experiencer, attitude, object), which relates the state(p) attitude of the experiencer on an object.

Sentiment Analysis is different from the emotion analysis, where in former concerns only emotions: “brief episodes of synchronized responses (sad, angry, joy, etc.)” [6], while the later concerns emotions and attitudes: “relatively enduring, effectively

colored beliefs, preferences, and predispositions towards objects or persons (love, hate, like, etc.)” [6]. The basic measure of sentiment is the polarity (estimation of positivity or negativity in the given text). The simplest form of the polarity is to have only two degrees either positive or negative: either like or dislike button in Facebook, we can consider them as the extreme ends of the continuous scale. The continuous scale can have some discrete points, which result in rating scales such as Amazon rating scale and IMDB rating scale. Polarity is mapped in the scale range of  $[-1,1]$  whereas ‘-1’ is considered as negative, ‘1’ considered as positive and ‘0’ considered as neutral. Many researchers neglect the ‘0’ case as it is very tough even for human access, and analysts were forced to remove the data in this case [7] [8] [9].

### ***2.1.1 TYPES OF SENTIMENT ANALYSIS:***

Sentiment Analysis can be done at different levels, and these levels are the basis for these classifications. On the top level perspective, there are three types of sentiment analysis.

They are:

- Document Level Sentiment Analysis.
- Sentence Level Sentiment Analysis.
- Aspect Level Sentiment Analysis.

#### ***2.1.1.A Document Level Sentiment Analysis:***

The primary aim of this analysis is to determine whether the whole document is expressing positive, negative or neutral opinion about an entity. The main goal is to

extract the overall polarity of the document. This can be considered as a standard text classification problem and can be addressed by machine learning algorithms like Maximum Entropy and Naive Bayes Classifier [10]. This document type analysis is only applicable to cases where the author represents only a single entity such as movie reviews, hotels, and restaurants. There are several assumptions needed to be considered in this process.

- The document should be targeted on a single entity like a restaurant or a movie
- The author should be the opinion holder

This type of sentiment analysis may not be fit well to the cases where sentiment is expressed in blogs or forums: where the author expresses the sentiment by comparing various products in that same domain using some standard comparing words.

#### ***2.1.1.B Sentence Level Sentiment Analysis:***

The primary aim of this analysis is to determine whether the particular sentence is expressing a positive, negative or neutral opinion. This can also be considered as a standard text classification problem. This process can be achieved in two stages.

- Looking for Subjective sentences: In the whole bunch of document, the first task is to filter out the subjective sentences from the document leaving behind the objective sentences.
- Sentiment classification at sentence level: Once the subjective sentences are separated out, the next step is to determine the polarity expressed by the individual subjective sentence on a particular feature or object.



This type of sentiment analysis is well addressed by machine learning algorithms like Naive Bayes, decision trees, support vector machines and some ensemble methods.

#### ***2.1.1.C Aspect Level Sentiment Analysis:***

The main aim of this sentiment analysis is to determine the sentiment of individual sentences and phrases about a particular aspect. This type of sentiment analysis is also called as fine-grained sentiment analysis [11], where all the sentiment expressed by the aspect is considered and then makes a forecast. This type of sentiment analysis does not concentrate on a single linguistic entity. Instead, it focuses on the overall sentiment expressed on a particular aspect. Many Researchers [12] argues that ‘Aspect level Sentiment Analysis’ is the head of all sentiment analysis as it involves the establishment of relationships between the opinion holder and objects, which results in an efficient result. However, all these tasks urge for more advanced machine learning algorithms (ensemble methods) where research is already going on for developing efficient techniques.

#### ***2.1.2 SENTIMENT ANALYSIS PROCESS:***

The basis of sentiment analysis lies in identifying objects, features, opinion holder, phrases or words expressed on the particular object/feature which indicates the polarity orientation. To understand deeply about the process of sentiment analysis let’s make some assumptions. Suppose there is document with a set of sentences ‘D’

$$D = \langle s_1, s_2, \dots, s_m \rangle$$

which contains opinions on a set of objects ‘O’

$$O = \{o_1, o_2, \dots, o_n\}$$

expressed by a set of opinion holders ‘H’

$$H = \{h_1, h_2, \dots, h_q\}$$

Again each object is described by a set of features ‘F’

$$F = \{f_1, f_2, \dots, f_r\}$$

and each feature  $f_i$  is attributed by a set of phrase words ‘ $W_i$ ’ (synonyms of features)

$$W = \{w_{i1}, w_{i2}, \dots, w_{in}\}$$

or can be described by a set of feature indicators

$$I = \{i_{i1}, i_{i2}, \dots, i_{im}\}$$

A document ‘d’ can contain opinions on different objects, and each object can have many features associated with it. The opinion on each feature or an object is given by a set of adjectives or adverbs which are called phrase words or feature indicators.

There are two types of opinions extracted from the documents i.e. either direct opinion or a comparative opinion.

### ***2.1.2.A Direct Opinion:***

Direct opinion can be given by the quintuple  $(o_j, f_{jk}, oo_{ijkl}, h_i, t_l)$  where

‘ $o_j$ ’ represents a particular object in the document

‘ $f_{jk}$ ’ represents a particular feature associated with that object

‘ $h_i$ ’ represents a opinion holder, i.e, the author of the document

‘ $oo_{ijkl}$ ’ represents the polarity orientation related to the particular feature of an object, expressed by a particular opinion holder.

The opinion orientation can be positive, negative or it can take any scale measurement value. The opinion or a sentiment on a particular feature  $f_i$  given by the opinion holder  $h_i$  has been characterized by the usage of words with from the set 'W' or 'I'.

#### ***2.1.2.B Comparative Opinion:***

Comparative opinions are present in blogs or forums: where the opinion holder expresses his or her opinion by comparing two objects. The author may or may not use comparative or superlative words during this process. This process involves establishing a relationship between the objects of different features and extracting the underlying sentiment.

## **2.2 MACHINE LEARNING**

Arther Samuel, the pioneer in the field of Artificial Intelligence, is the one who coined the term Machine Learning in the year 1959. He defined Machine Learning as “A field of study that gives the computers ability to learn without being explicitly programmed” [13]. However, there are many different definitions put forward after that. In a generalized sense, one can understand Machine Learning as the process of making a machine to learn from the given situations/instances through different algorithms and enabling it to perform on its own in the future through the acquired knowledge/intelligence. Machine Learning is considered as the core field of Artificial Intelligence, which came into a picture by the study of computational linguistics, pattern

recognition and science [14], it claimed to be extracting many underlying principles from computer science and statistics.

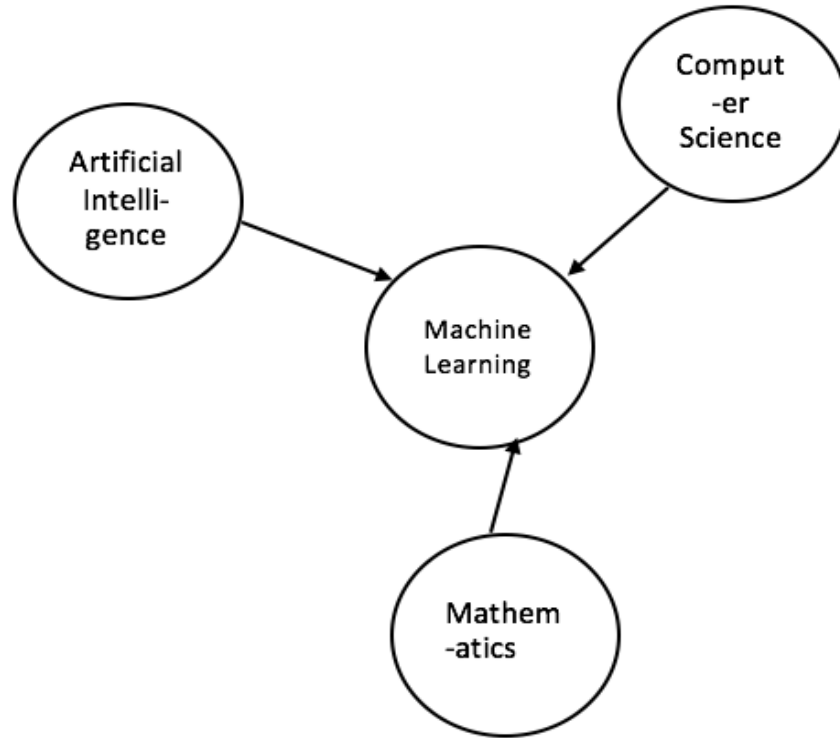


Figure 2.1: Evolution of Machine Learning

### ***2.2.1 DATA ORIENTED MACHINE LEARNING:***

In data point of view, Machine Learning can be defined as the ability of the machine to learn from the given data using statistically programmed algorithms and predict the future data. It is almost like devising a new complex model built with the given sample data and using the model to predict the future target function, typically referred as Predictive Analytics.

### ***2.2.1.A Learning Process:***

A typical machine learning process involves training from the given input data, developing a hypothesis model and predicts the future variable. Generally, in a data-driven machine learning process, the given raw data is divided into two sets, training set and test set. Using the training set, we will develop a final hypothesis and test set is used as a new data to test the performance of the final model/hypothesis.

In any training process, the final output model should generalize the data [15][16] rather than memorizing it. Depending on the training and test accuracy, one can say whether the model is generalized or memorized the data. If the test set accuracy is much lower than the training set, it is considered that the model memorizes the data in the training process (over-fitting condition), if test accuracy is slightly varied with the training accuracy, then the model is developed by generalizing the data in the training process. In model generalization process, we need to consider the underfitting case, if the model is too generalized than necessary, the model may underfit the data.

The set of all the equations that we fit the training data is referred to as the hypothesis space. The one best-fitted equation which reduces the risk or cost on the overall training data, is taken as the Final Hypothesis model. Generally, the final hypothesis model is taken by different methods like maximum likelihood estimation or Empirical Risk Minimization.

The complete Loss or Cost or Risk Function is given by

$$R_D(h_{i(x)}) = \frac{1}{N} \sum (y_i - h_1(x_i))^2$$

where  $R_D$  is the Risk function over the Data 'D' having 'N' data points.

$y_i$  is the observed value (target or label) and  $h_1(x_i)$  is the predicted value.

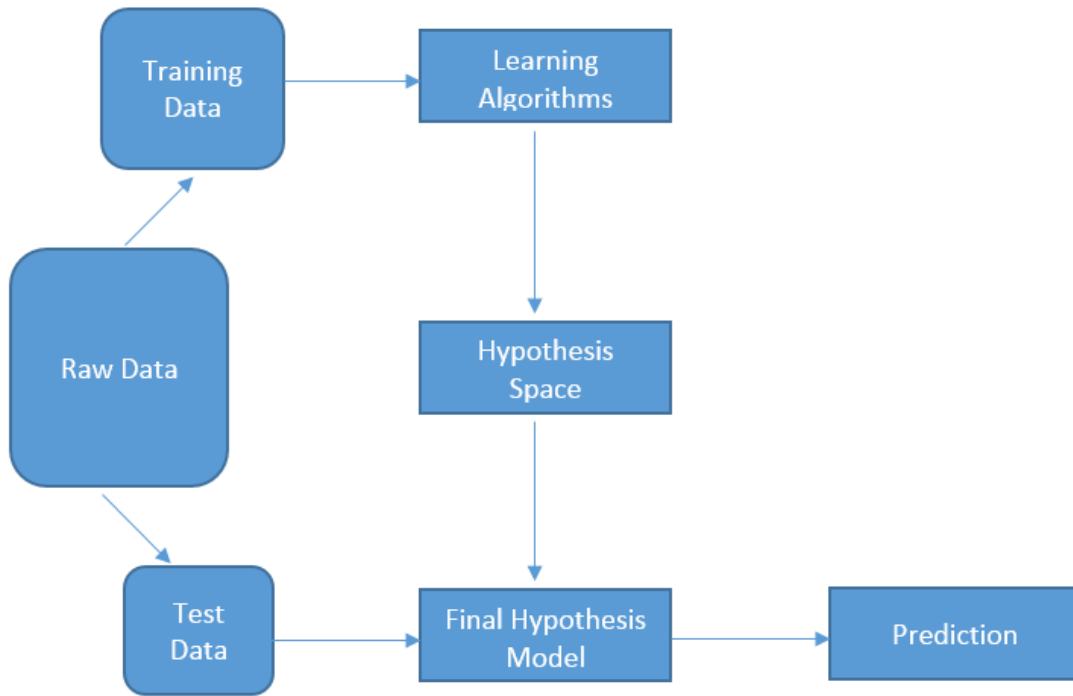


Figure 2.2: Machine Learning Process

Now the final hypothesis ‘g’ is the one, who is having lowest risk function in the whole hypothesis space ‘H’ (represents all the polynomial spaces)

$$g(x) = \operatorname{argmin}_{h(x) \in H} R_D(h(x))$$

So g(x) is the final hypothesis model which best fits the given data. During the training process, various considerations are to be taken for bias and variance just to make sure that the curve is not overfitting or underfitting the data.

### ***2.2.2 SUPERVISED VS UNSUPERVISED:***

Supervised Learning is the one, where the given data is the labeled, and the learning process tries to find a mapping function from the given inputs to the given outputs.

#### ***2.2.2.A Regression vs Classification:***

If the target values are continuous, it is called a Regression Problem. The various Regression problems include linear regression, ridge regression, logistic regression, etc. If the target values are discrete, then it called a classification problem, and the common classification algorithms are Simple Vector Machines, KNN's, Decision Trees, Naive Bayes, etc.

Unsupervised Learning is the one, where the given data is unlabeled, and the machine should itself find out the hidden patterns in the given data. All the unsupervised problems are related to classification task. Common unsupervised tasks include KNN's clustering, etc.

### ***2.2.3 PARAMETRIC VS NONPARAMETRIC:***

Parametric learning works on the principle of assumptions, i.e. the algorithms themselves simplify to a function of known form depending on different parameters such as mean, median, and mode. In nonparametric learning, algorithms do not consider any assumptions, and they have the freedom to choose any form of the function. Non-parametric learning methods are much accurate and more flexible while parametric learning methods are fast [38] [39].

	<b>Supervised Learning Methods</b>	<b>Unsupervised Learning Methods</b>
<b>Parametric Learning Methods</b>	Linear Regression, Logistic Regression, Linear Discriminant Analysis, Perceptron, Naïve Bayes, Simple Neural Networks	Cluster Analysis
<b>Non-Parametric Learning Methods</b>	Support Vector Machines, KNN, Decision Trees, Ensemble Methods(Bagging, Boosting, etc.)	KNN, Cluster Analysis, Mean shift

Table 2.1: Classification of Learning algorithms

#### **2.2.4 NAIVE BAYES ALGORITHM:**

Naive Bayes algorithms is used for classification problems. It is a full probabilistic model based on Bayes theorem [37].

#### **Bayes Theorem:**

Bayes theorem states that the posterior probability distribution of ‘A’ is equal to the likelihood times( $P(B/A)$ ) the prior probability distribution of ‘A’.



$$P\left(\frac{A}{B}\right) = \frac{P\left(\frac{B}{A}\right) P(A)}{P(B)}$$

Here  $P\left(\frac{B}{A}\right)$  represent the likelihood of 'B' given 'A', lets represent that as  $L(A)$ . Then the above equation can be written as

$$P\left(\frac{A}{B}\right) \propto L(A) * P(A)$$

Suppose  $A = Y$  (Parameter of data)

$B = X$  (data)

then the above proportionality changes to

$$P\left(\frac{Y}{X}\right) \propto L(Y) * P(Y)$$

i.e. the posterior distribution of the data is equal to the likelihood of the data for the given parameter times the prior probability of the parameter.

Prior probabilities require domain knowledge, and in many cases, people take an arbitrary value. Likelihood is calculated on the new data, i.e. probability of the new data for a given parameter. When it comes to Bayesian modeling sample size matter a lot, if the sample size is too small then prior dominates the likelihood and the result will be dependent on the prior rather than the new data. So sample size and prior probabilities are crucial in Naive Bayes algorithm.

Naive Bayes, as the name indicates, it is very naive. It depends on the conditional independence of events. For a better understanding of Naive Bayes, let's consider the Naive Bayes spam filter.

Assume that there are five words frequently appear in the spam mail and suppose that someone got a new mail which uses three words of the spam category and rest are not, then the probability of the given mail being spam is given by

$$P\left(\frac{spam}{W_1, W_2^c, W_3, W_4^c, W_5}\right) = \frac{P\left(\frac{W_1, W_2^c, W_3, W_4^c, W_5}{spam}\right) P(spam)}{P(W_1, W_2^c, W_3, W_4^c, W_5)}$$

Now by Naive Bayes assumption, let's consider the conditional independence of words in the spam and conditional independence of words in the non-spam. Then we can write the likelihood as

$$P\left(\frac{W_1, W_2^c, W_3, W_4^c, W_5}{spam}\right) = P\left(\frac{W_1}{spam}\right) * P\left(\frac{W_2^c}{spam}\right) \dots P\left(\frac{W_5}{spam}\right)$$

This might be a huge assumption, but it makes the computation much simpler.

Similarly, the not spam likelihood is given by the below formula:

$$P\left(\frac{W_1, W_2^c, W_3, W_4^c, W_5}{not\ spam}\right) = P\left(\frac{W_1}{not\ spam}\right) * P\left(\frac{W_2^c}{not\ spam}\right) \dots P\left(\frac{W_5}{not\ spam}\right)$$

Sometimes, given the time and space complexity, it is useful to use addition instead

of multiplication. So by applying log to the above discussed equations, multiplication can be changed to addition.

$$\log(P(A) * P(B)) = \log(P(A)) + \log(P(B))$$

### **2.2.5 METRICS OF MACHINE LEARNING:**

#### **Confusion Matrix:**

A tabular representation of a matrix with True Negatives, False Positives, False Negatives, and True Positives.

<b>Data</b>	<b>Prediction 1</b>	<b>Prediction 0</b>
<b>Observed 1</b>	True Positives	False Negatives
<b>Observed 0</b>	False Positives	True Negatives

Table 2.2: Confusion Matrix

#### **True Positive Rate:**

It is also termed as recall, i.e. it specifies the recalling capacity of the machine. It is the ratio of the true positives to the total positives in the given data.

$$TPR = \frac{TP}{TP + FN} = Recall$$

**False Positive Rate:**

It is the ratio of the false positives to the total negatives in the given data. It conveys the proportion of the miss classification of predicted positives.

$$FPR = \frac{FP}{FP + TN}$$

**Precision:**

It is the ratio of the true positives to the total predicted positives at the output. It defines the fraction of positive instances that are relevant to the input.

$$Precision = \frac{TP}{TP + FP}$$

**Accuracy:**

It is the total measure of how accurate the output is, i.e. it is the ratio of the sum of true positives and true negatives to the total population

$$Accuracy = \frac{TP + TN}{FN + TN + TP + FP}$$

**F-Score:**

For unbalanced sets, where accuracy score may be more biased, F-score can be used as a good metric. F- Score is the weighted harmonic mean of precision and recall.

$$F - score = \frac{Precision + Recall}{2 * Precision * Recall}$$

**Roc Curve:**

It is the curve plotted between TPR and FPR, with TPR on x-axis and FPR on y – axis. The more corner the curve is having at the northwest direction, the more accurate the output.

**Precision – Recall Curve:**

It is the curve plotted between Precision and Recall with Precision on the x-axis and Recall on the y-axis. For some problem, Precision-Recall curve makes more sense than the ROC curve.

## 2.3 BIG DATA

Although the term Big Data is famous from early 20's it has been in use since 1990. The credit goes to John Mashey, for coining the term Big Data [18]. It refers to the huge volumes of Datasets, which cannot be handled by a single machine or any normal traditional database system. Gartner, the world's leading IT advisory and research company, discussed this growth rate challenges and presented the scale as being as three-

dimensional i.e. volume, velocity, and variety [19]. Soon after its disposition, the 3V's concept became very famous they defined Big Data as “Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery, and process optimization.” Later this 3V's model has been expanded to 4V's model by some other organizations such as IBM, coining the 4th V as veracity [20].

### **2.3.1 ATTRIBUTES OF BIG DATA:**

The various attributes of the Big Data are:

1. Volume
2. Velocity
3. Variety
4. Veracity

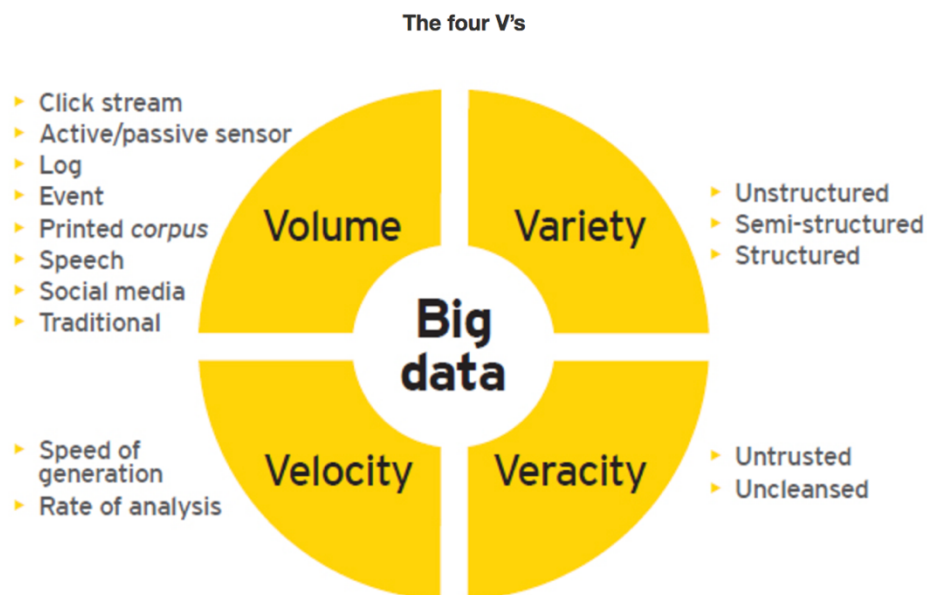


Figure 2.3: Attributes of Big Data [21]

### ***2.3.1.A Volume:***

Volume is the most important attribute of Big Data, which demands the scalable and distributed form of data storage resources. The amount of data has been increasing day by day, peta bytes to exa bytes, exa bytes to zeta bytes. Enterprises have large volumes of data, which have been storing for many years and the traditional warehouse devices are unable to process or store these massive amounts of data due to the lack of distributed and parallel processing architecture. Some resources for these huge volumes of data include:

- Log Records,
- Transactional data for consumer preferences,
- Spatial and temporal data, etc.

### ***2.3.1.B Velocity:***

It represents the speed at which the data needs to be handled. Due to increased customer insight requirements, web and mobile technologies are having a specific mechanism where most of the data is redirected to hosting providers. Imagine, how many messages, status updates, likes and dislikes, were generated by Facebook per minute, how many tweets are updated by users per minute in Twitter, how many Instagram photos, etc. All these situations impose a need for developing a system, which should handle this high-frequency data.

### ***2.3.1.C Variety:***

Due to the advancements in the technology, companies want to extract customer insights almost from any form of data. The various forms of data include structured, semi-structured and unstructured data. By augmenting the structure information which can fit any traditional relational database with the semi-structure and unstructured data, companies can able to get deeper insights about the customer.

### ***2.3.1.D Veracity:***

Veracity refers to the trustfulness of the data. Now a day's, data is being collected from many sources. Here comes the questions like,

How far can we trust the data?

How far can we take decisions depending upon the data?

How far the data cleaned?

Because a good scientist knows that there are subtle discrepancies underlying in all the data collected.

With all the different dimensions of the data mentioned above, there comes a need for specialized algorithms, tools and storage systems for processing that massive amount of data. For all mentioned dimensions, Big Data Analytics offers a wide range of tools for processing and storage of these enormous amounts of data.

### ***2.3.2 HADOOP:***

The field of Big Data offers many tools for processing and storage of these large amount of information. When talking about Big Data, it is hard to skip the role of Hadoop



in it. Hadoop is the open-source distribution of Apache Software Foundation. The Apache Foundation defined Hadoop as “A framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models [23].” It is designed to handle to huge amounts of data across a large number of machines with each machine having its own memory and processing power. The program is developed in such a way, the system will automatically handle fault tolerant and scalability issues rather than depending upon hardware availability.

#### ***2.3.2.A Hadoop Architecture:***

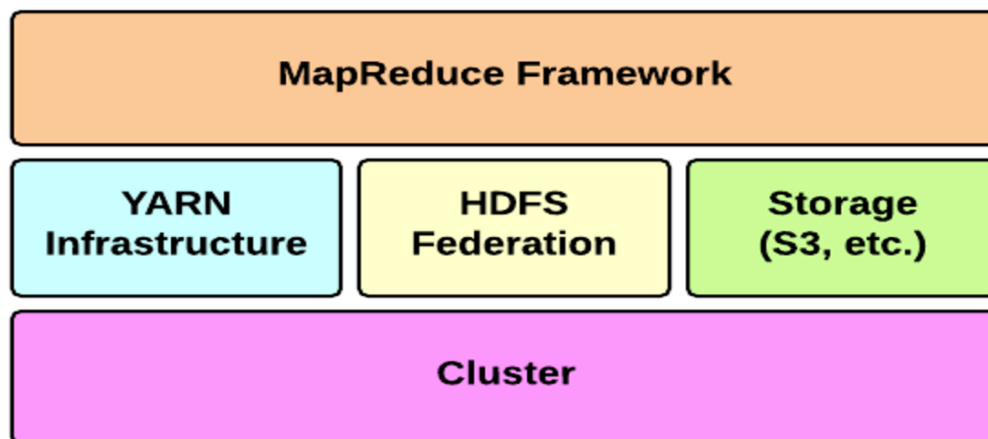


Figure 2.4: Hadoop Architecture [22]

The main components of Hadoop include:

#### **Map Reduce:**

It defines the processing software framework developed by Google for parallel processing of large scale of information. It consists of two phases map, and reduce phase.

At map phase, data nodes collect the data from HDFS, perform mapping operations locally. At reduce phase, nodes collect these locally stored data, perform the final reduce operations and store the results back into HDFS.

### **HDFS:**

HDFS stands for Hadoop Distributed File System. It has a master-slave configuration. This file system is distributed across all the nodes making the data available to every node in the cluster. Data is stored in the form of duplication factors as in redundancy format just in case to handle any fault tolerant issues.

### **Cluster:**

A Hadoop cluster is nothing but a group of nodes aligned in the form of a master-slave configuration. Again, these nodes are partitioned into racks. Generally, cluster represents the hardware infrastructure aspect of Hadoop.

### **YARN:**

YARN stands for Yet Another Resource Negotiator. It came into a picture from Hadoop 2.0. It is considered as the operating system of the Hadoop, acting as a resource-management framework. It is responsible for assigning and managing the computational resources across all the nodes in the cluster which are needed for the execution of different applications.

Spark and most other big data tools are developed on the Hadoop concept of distributed computing and the resource manager.

## **2.4 SUMMARY**

This chapter provides all the domain knowledge and the literature review for the thesis, which helps in good understanding of the proposed framework and its implementation. Here, we started by giving details on sentiment analysis, followed by Machine Learning and ended up with Big Data Analytics. In the next chapter, we will provide information about the tools used in developing the framework.

## **CHAPTER – 3**

### **TOOLS BACKGROUND**

In this thesis, the proposed framework has to handle different case studies with different problem situations. For developing the framework, we used different tools in order to accomplish different tasks. This part emphasizes on the tools used in developing the proposed framework.

#### **3.1 PROFILE OF TOOLS ARCHITECTURE**

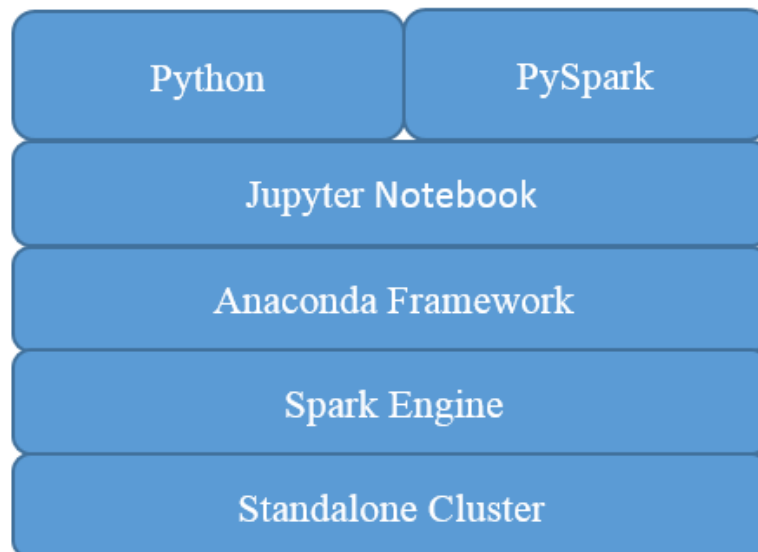


Figure 3.1: Profile of Tools Architecture

## 3.2 ANACONDA FRAMEWORK

Anaconda is the product of Continuum Analytics, they define anaconda as “A free, easy-to-install package manager, environment manager, Python distribution, and collection of over 720 open source packages with free community support [26].”

Anaconda comes with two different working modules, anaconda navigator, and conda.

### 3.2.1 ANACONDA NAVIGATOR:

Anaconda Navigator is a free easy to use GUI interface offer by Anaconda which helps in launching different applications, managing conda packages, channels, and environments without the use of command line interface [27].

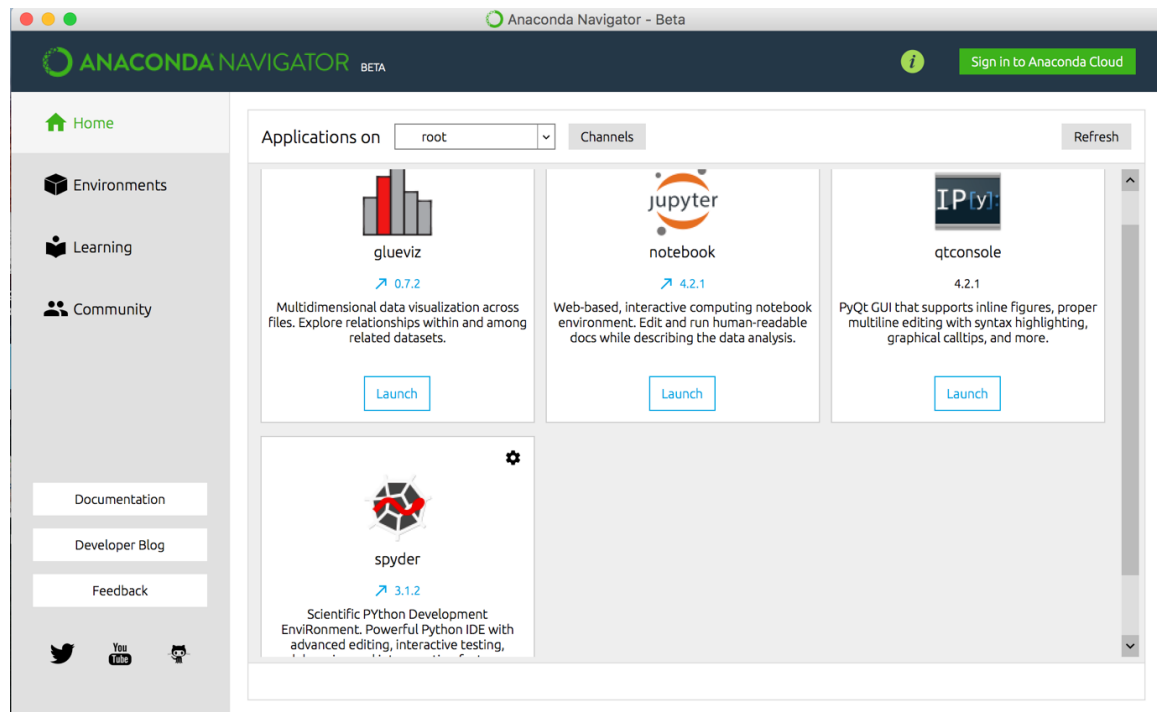


Figure 3.2: Anaconda Navigator

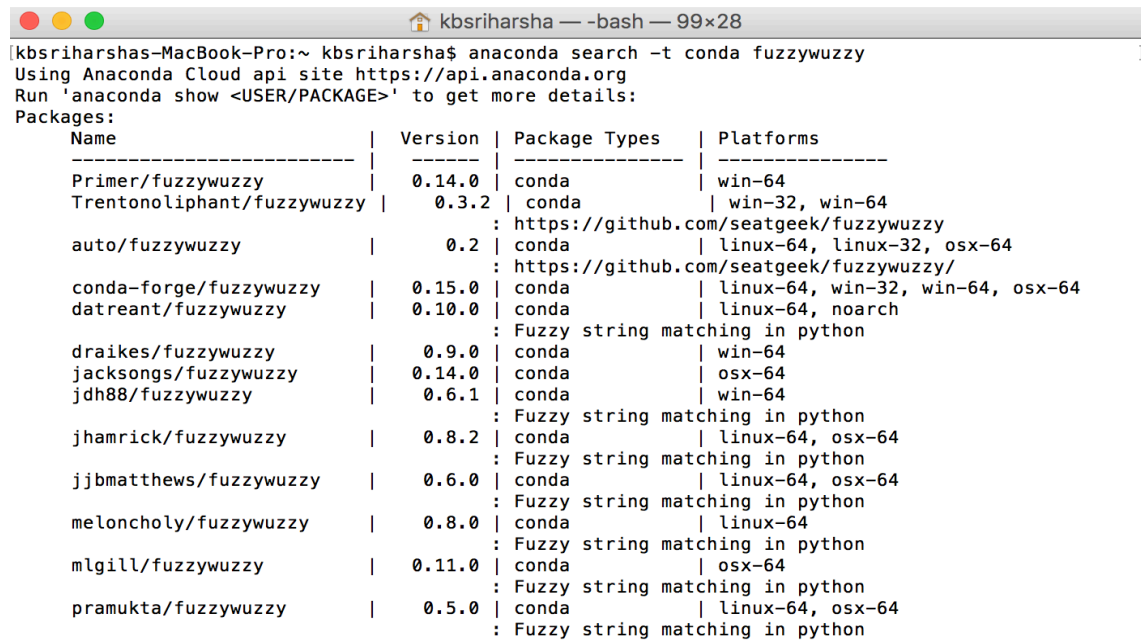
### 3.2.2 CONDA:

Conda offers a command line interface to manage packages, environments, and channels [28]. It can be used on any operating system such as OSX, Linux, and Windows. Generally, during installation anaconda comes with 150 scientific packages of python and additional 250 packages can be installed.

#### To install a package:

`conda install <package-name>`

Sometimes anaconda unable to find the channel for downloading the specified package. In that cases, one has to search for the packages offered by various channels and then install from one of the specified channels according to our system specifications.



```
kbsriharsha — -bash — 99x28
kbsriharsha-MacBook-Pro:~ kbsriharsha$ anaconda search -t conda fuzzywuzzy
Using Anaconda Cloud api site https://api.anaconda.org
Run 'anaconda show <USER/PACKAGE>' to get more details:
Packages:
  Name | Version | Package Types | Platforms
  -----|-----|-----|-----
  Primer/fuzzywuzzy | 0.14.0 | conda | win-64
  Trentonoliphant/fuzzywuzzy | 0.3.2 | conda | win-32, win-64
  : https://github.com/seatgeek/fuzzywuzzy
  auto/fuzzywuzzy | 0.2 | conda | linux-64, linux-32, osx-64
  : https://github.com/seatgeek/fuzzywuzzy/
  conda-forge/fuzzywuzzy | 0.15.0 | conda | linux-64, win-32, win-64, osx-64
  datreant/fuzzywuzzy | 0.10.0 | conda | linux-64, noarch
  : Fuzzy string matching in python
  draikes/fuzzywuzzy | 0.9.0 | conda | win-64
  jacksongs/fuzzywuzzy | 0.14.0 | conda | osx-64
  jdh88/fuzzywuzzy | 0.6.1 | conda | win-64
  : Fuzzy string matching in python
  jhamrick/fuzzywuzzy | 0.8.2 | conda | linux-64, osx-64
  : Fuzzy string matching in python
  jjbmatthews/fuzzywuzzy | 0.6.0 | conda | linux-64, osx-64
  : Fuzzy string matching in python
  meloncholy/fuzzywuzzy | 0.8.0 | conda | linux-64
  : Fuzzy string matching in python
  mlgill/fuzzywuzzy | 0.11.0 | conda | osx-64
  : Fuzzy string matching in python
  pramukta/fuzzywuzzy | 0.5.0 | conda | linux-64, osx-64
  : Fuzzy string matching in python
```

Figure 3.3: Command Interface of Conda

**To install package from a specified channel:**

*conda install -c <channel-name> <package-name>*

Anaconda also offers various means for creating different environments, for scalable computing, etc.

**Packages Used in the framework:**

- tweepy
- matplotlib
- plotly
- pandas
- json
- numpy
- os
- re
- nltk
- sklearn

These are the few mentions of the packages used in developing this thesis.

However, the actual framework uses 15 – 17 packages.

### 3.3 JUPYTER NOTEBOOK

“Jupyter Notebook” can be considered as a web application or interactive computing environment that allows to create and share documents that contain live code, equations, visualizations and explanatory part [29] [30].

The main components of the jupyter notebook are notebook web application, kernels, notebook documents.

- Notebook Web Application: Notebook web application provides an interactive platform for writing code, HTML, markdown, and to visualize results.
- Kernels: These are involved in the default startup process in any Notebook application, that runs the code written in the cells and returns the result back to the cell.
- Notebook Documents: These are the self-contained documents that contain all the things visualized in the notebook such as code, input, output, HTML, and markdown [30].

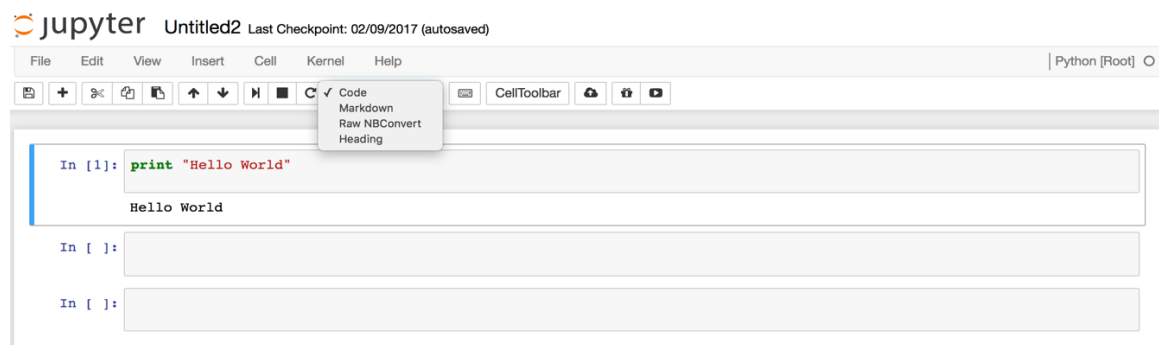


Figure 3.4: A View of Jupyter Notebook by highlighting different ways to write



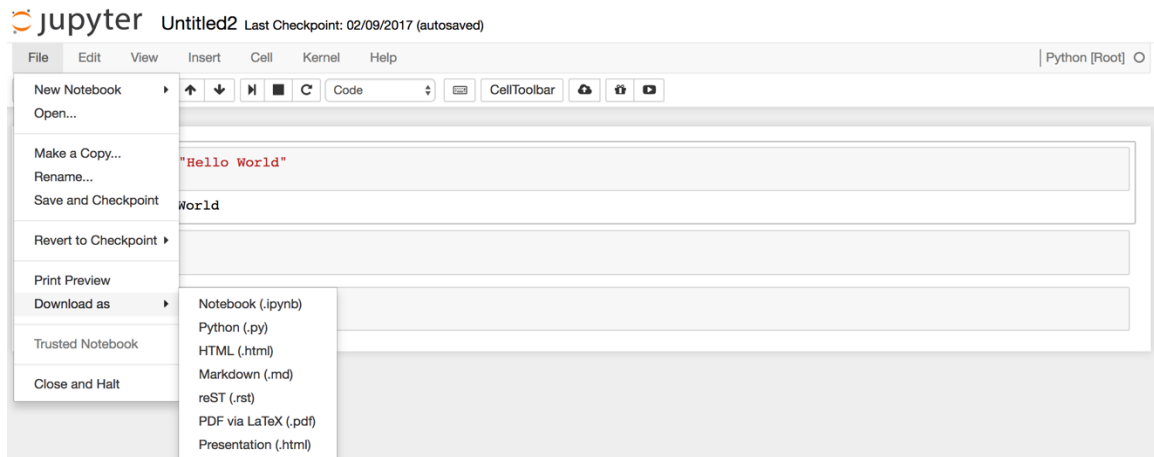


Figure 3.5: Notebook by highlighting different downloadable options

## Complete Notebook:

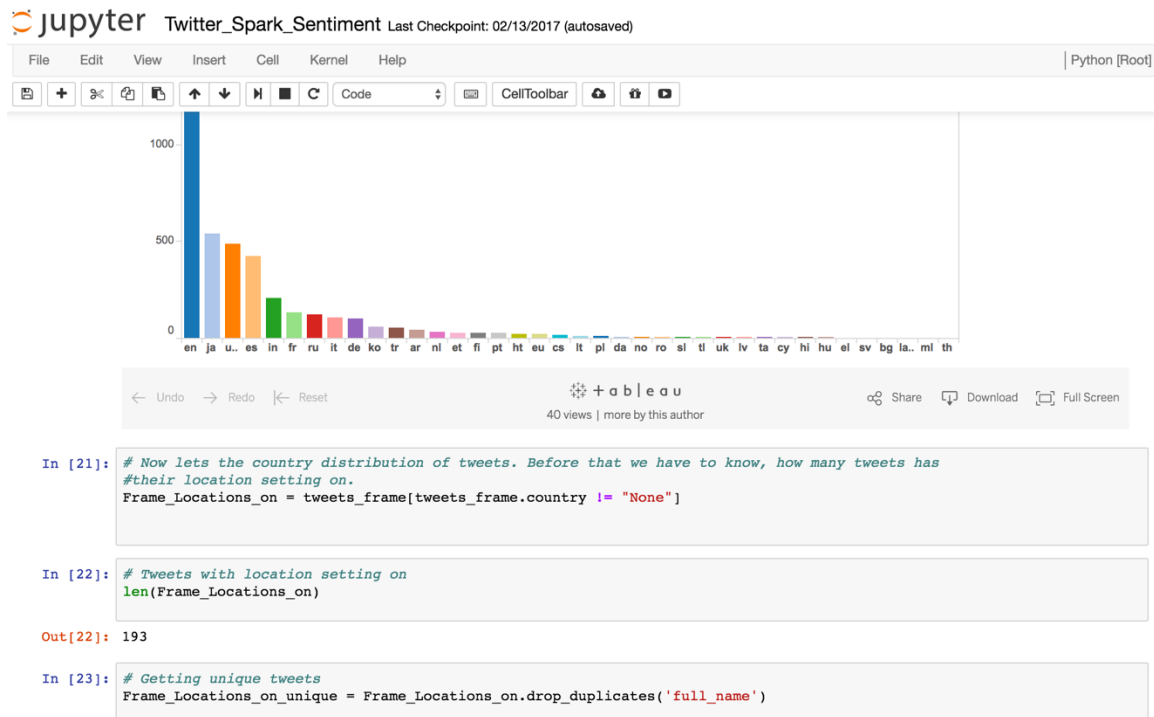


Figure 3.6: Sample of a complete Notebook

So Jupyter Notebook is an all in one environment for the developmental purposes. It will give a data scientist more convenience with its high level of easiness and customization.

### **3.4 SPARK**

Spark is a general-purpose large-scale cluster-computing framework [34] with many data tools stacked under a single umbrella for handling different cases. One of the most important aspects of spark is its scalability and efficiency. It is well built with a higher level API's which are available in different programming languages like Java, python, and scala. Data operations in spark are performed by transformations and actions. Unlike most frameworks, spark can disseminate these operations across various worker nodes by providing programming abstraction and speed by parallel processing.

#### ***3.4.1 DATA ABSTRACTIONS IN SPARK:***

The two important data abstractions in Spark are RDD's and Data Frames.

##### **RDD:**

RDD stands for Resilient Distributed Dataset, and they are the primary abstraction in spark. An RDD is the immutable distributed collection of the data across all the nodes in the cluster [33]. It operates in parallel using lower-level API's. Generally, RDD's are preferred when data is unstructured or when somebody want to work at the core abstraction of spark.

### DataFrames:

Like an RDD, DataFrame can also be considered as an immutable distributed dataset which assigns a schema to the data by making processes much easier. It can be equivalent to a relational table in RDBMS. They are entangled with more optimizations such as catalyst optimization, which offers much more performance. It offers a high-level abstraction and makes its accessible to common dataframe language like SQL.

### 3.4.2 ARCHITECTURE OF SPARK:

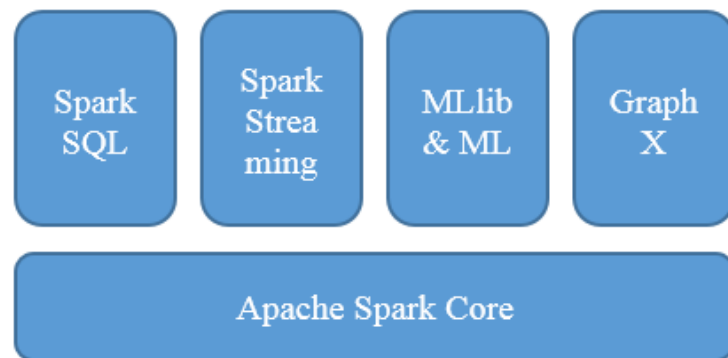


Figure 3.7: Architecture of Spark

### Spark SQL:

Spark SQL provides SQL type writing to the Data Frames. SQLContext is the entry point for Spark SQL. SQLContext is created from SparkContext, which will be created at the start of an application.

**Spark Streaming:**

Spark Streaming is used for live processing of streaming data by offering high throughput and fault-tolerance. [31] The input sources can be anything such as Twitter, Kafka and Flume.

**MLlib & ML:**

Both MLlib and ML are the high-level machine learning API's offered by the spark. MLlib is based on RDD's, while ML is based on DataFrames. Both API's provides different methods for ML purpose and featurization. They also provide an ML pipeline which is nothing but a pipelined series of estimators and transformations.

**GraphX:**

GraphX is an RDD based API which offers parallel computation through graphs. GraphX is still maturing with new algorithms and builders [32].

In the development of this framework, Spark was used with both RDD and DataFrame abstraction.

### 3.4.3 SPARK WORKFLOW IN PROPOSED FRAMEWORK:

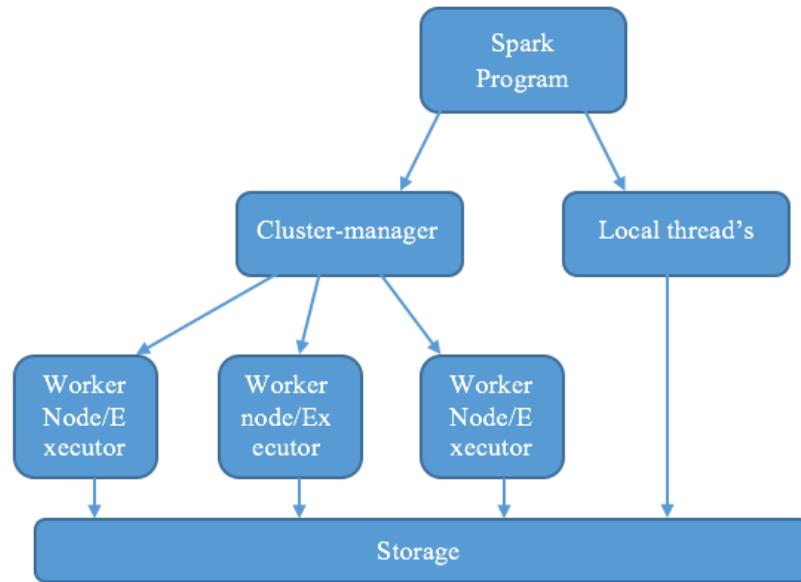


Figure 3.8: Spark workflow in Proposed Framework

A spark program consists of two parts, driver program and worker program. The driver program is responsible for creating the distributed datasets across the nodes of the cluster and then performs the transformations and actions on these datasets across the nodes. Worker program runs on the executor or worker nodes. For allocating resources to executors and for running these programs spark consists of a cluster management tool in its architecture, which can be anything such as Apache Mesos, YARN, and Spark Standalone cluster.

In the present framework, Spark operates in Standalone cluster mode with two local threads. Here, in this case, PySpark is the driver program and worker program runs on the local threads. Initially, the application starts by creating the SparkContext object, which is the main entry point for a spark application. Then SQLContext is created from

the SparkContext which is the main entry point for the spark DataFrames and its operations. SparkContext and SQLContext are initiated by the notebook when using Jupyter Notebook and for rest of the tools, these have to be initiated by the programmer.

```
In [1]: # In ipython notebook SPARK CONTEXT is already initiated with a variable 'sc' and SQL CONTEXT is initiated as 'sqlContext'
print sc
print sqlContext

<pyspark.context.SparkContext object at 0x1099793d0>
<pyspark.sql.context.HiveContext object at 0x111eb7090>
```

Figure 3.9: sc and sqlContext variables in Jupyter Notebook

### 3.5 SUMMARY

This chapter serves the purpose of providing the documentation for the tools used in developing the proposed framework. We started by explaining the profile architecture of tools in the proposed framework and then followed by giving details on each tool. In the next chapter, we will explain the system architecture and implementation of the proposed framework.

## **CHAPTER – 4**

### **SYSTEM ARCHITECTURE AND IMPLEMENTATION**

This chapter focuses on the actual architecture and implementation of the proposed framework by presenting architectural block diagram and the process flow implementation charts.

#### **4.1 TOP LEVEL SYSTEM ARCHITECTURE OF PROPOSED FRAMEWORK**

The architectural block diagram of the proposed framework will consist of a data source, which is a social networking site (Twitter), Exploratory Analysis Section, Modelling Section, and local storage.

Figure 4.1 represents the architectural block diagram of the proposed framework with all the components for input, processing, and output. Twitter Streaming is used to collect tweets from the Twitter, and the raw data of the collected tweets is stored in the local database. Data Exploration section is the preprocessing and analysis section, where the collected data is subjected to various cleaning and preparation steps to make it more human readable and ready for modeling part. The final cleaned output is sent to Visualization section, which shows some exploratory analysis with interactive visualizations using Tableau. Now the cleaned data is collected from the local storage of our system and sent to the spark engine to perform the predictive sentiment analysis. The

final Output of the sentiment is visualized using Tableau plots. The outputs of every stage are stored in the local database for efficiency and documentation purposes.

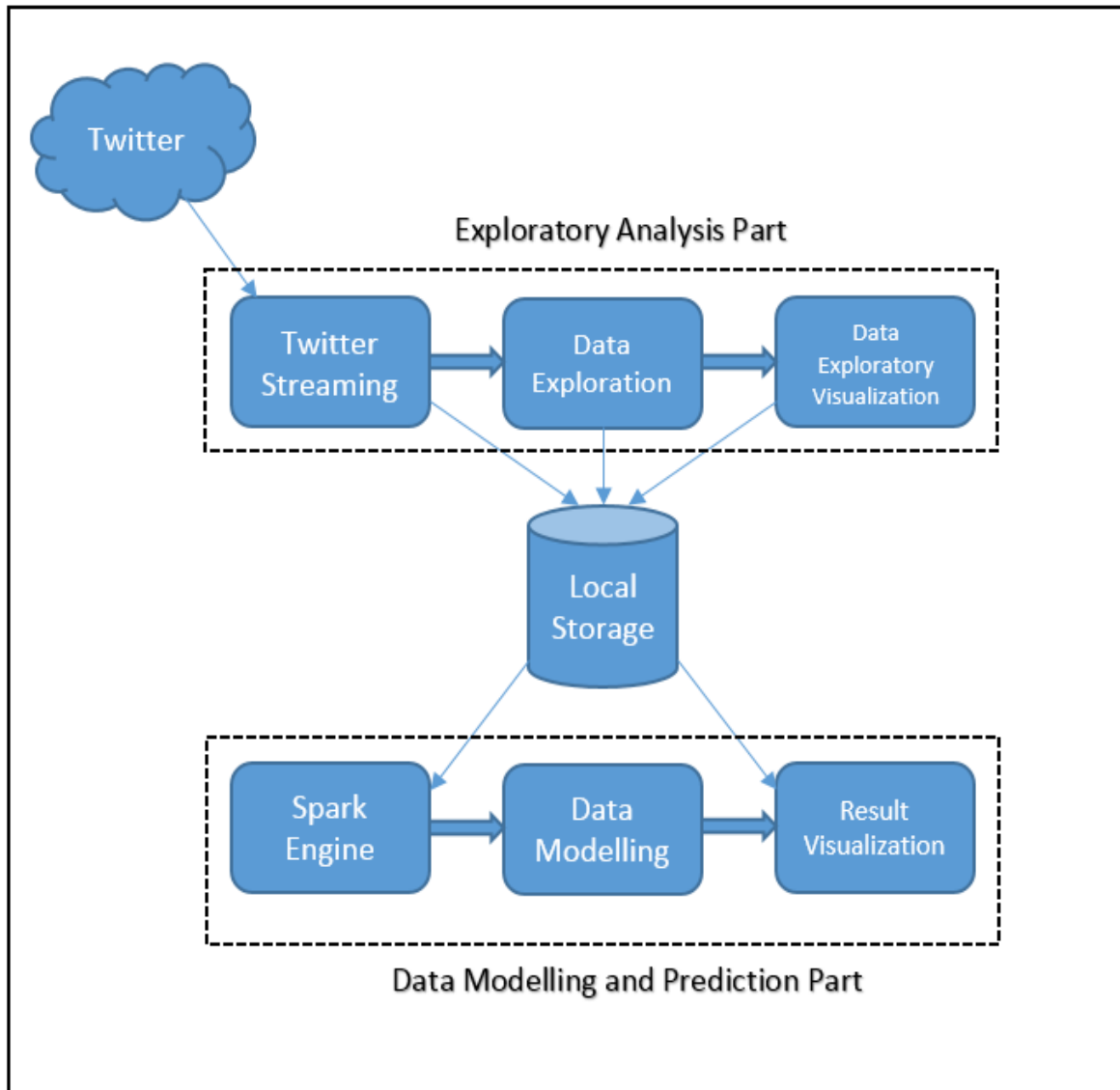


Figure 4.1: Top Level Architecture of the Framework



## 4.2 PROCESS FLOW

Figure 4.2 represents the process flow for the proposed framework which starts with data collection from Twitter, then preprocessing using the linguistic rules such as tokenizing, stemming, and lemmatizing, then followed by Data Modelling, which performs sentiment analysis and finally ends with the Results section.

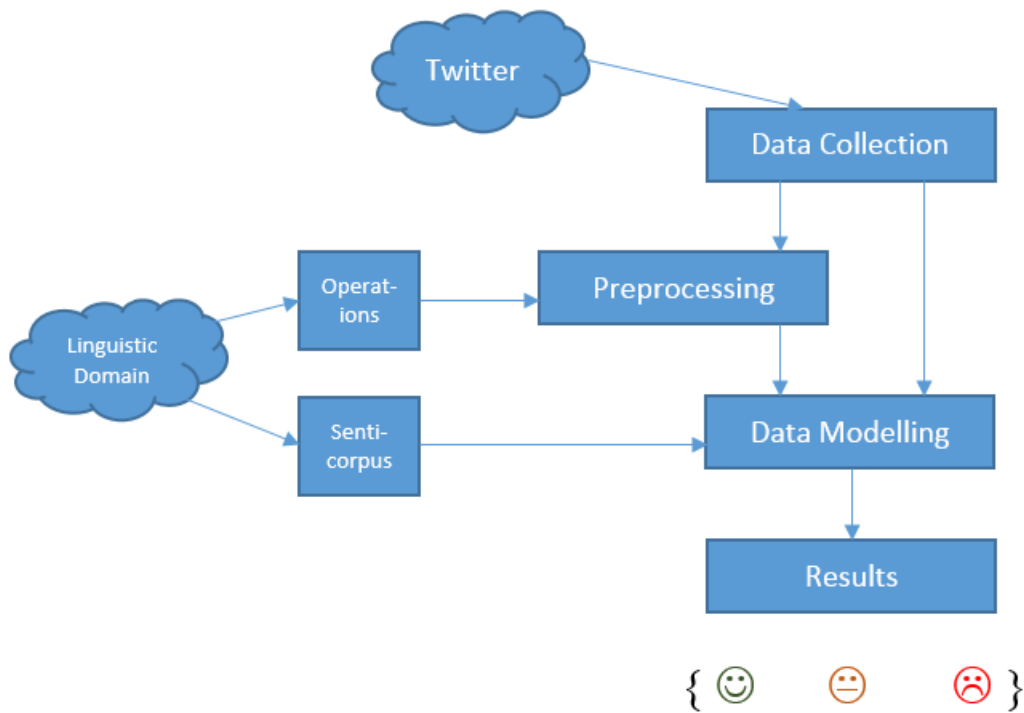


Figure 4.2: Process Flow in Proposed Framework

### 4.2.1 DATA COLLECTION:

The input data source here is the tweets, and Twitter is the data repository. For data collection purposes, Twitter provides two API's, streaming API and REST API. Streaming API is used to collect tweets in real-time, whereas REST API provides the way for collecting tweets from the past week. Streaming API offers longer connectivity

and provides data in almost real-time. Twitter search API is one of the REST API's of Twitter which almost equal to Twitter search functionality. The returned data is in the form of JSON or XML using GET requests. In most cases, the returned format will be in JSON, which is having more compactness.

The returned tweet consists of various fields like

```
tweet_fields = [u'contributors', u'coordinates', u'created_at', u'entities',  
                u'favorite_count', u'favorited', u'filter_level', u'geo', u'id',  
                u'id_str', u'in_reply_to_screen_name', u'in_reply_to_status_id',  
                u'in_reply_to_status_id_str', u'in_reply_to_user_id', u'in_reply_to_user_id_str',  
                u'lang', u'place', u'retweet_count', u'retweeted', u'source',  
                u'text', u'timestamp_ms', u'truncated', u'user']
```

Figure 4.3: Attributes of the returned tweet

The field 'tweet' contains the actual tweet posted by the user. So, the tweet field of every returned tweet is collected and sent to the preprocessing section.

In the implementation process, the proposed framework is being implemented with two use cases. In the first use case, Twitter streaming API was used and in the second use case, Twitter search API was used. In both cases, we can pass a specific word on which the tweets have to be collected. In Twitter streaming, the keyword is called filter, and in search API it is called query word.

#### 4.2.2 PREPROCESSING:

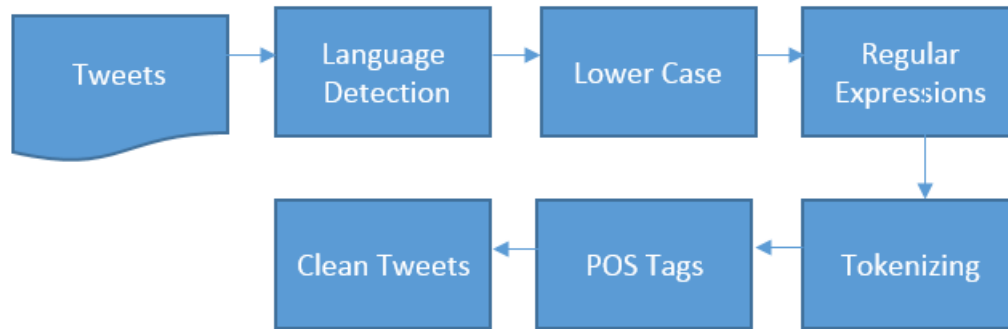


Figure 4.4: Preprocessing

##### **Language Detection:**

The tweets collected from Twitter streaming will contain tweets with different languages from various countries. Therefore, the first step in our process should be to separate English language tweets from the collected Twitter corpus.

##### **Lower Case:**

One way to make processes easier is to normalize the tweets. In this case, the tweets are normalized by converting them all to lower case which makes it easier for comparing the with the sentiment dictionary.

##### **Regular Expressions:**

Regular Expressions are used to select a particular type of content according to the pattern given in the expression. In this case, regular expressions are used to remove punctuation, special character (except space).

**Tokenization:**

Tokenizing is the process of breaking a sentence by the white spaces or any kind of special symbols. Each meaning word or symbol is called a token. Here tokenization was done by white spaces.

**Parts of Speech Tagging:**

POS tagging is the process of assigning parts of speech tag to every word in the sentence. We used POS tagging to detect the subjective words from the tweets like the like adjectives, adverbs and verbs.

**4.2.3 DATA MODELLING:**

Once tweets are cleaned the next step is the data modeling part, i.e. the sentiment assignment part. Here for sentiment polarity assignment, Naive Bayes theorem was used.

Initially, a word dictionary is maintained with all the positive and negative sentiment words with their log probabilities. When a new tweet is given the individual words are identified in negative and positive log probabilities dictionaries, and final negative log probability and positive log probability is calculated which is the sum of their individual log probability values. Final polarity is assigned according to the winning polarity log probability.

Suppose there are five subjective words  $\{w_1, w_2, w_3, w_4, w_5\}$  in the tweet.

Total Negative Log Probability =

$$\text{Neg\_Log\_Prob}(w_1) + \text{Neg\_Log\_Prob}(w_2) + \text{Neg\_Log\_Prob}(w_3) + \text{Neg\_Log\_Prob}(w_4) + \text{Neg\_Log\_Prob}(w_5)$$

Total Positive Log Probability =

$$\text{Pos\_Log\_Prob}(w_1) + \text{Pos\_Log\_Prob}(w_2) + \text{Pos\_Log\_Prob}(w_3) + \text{Pos\_Log\_Prob}(w_4) + \text{Pos\_Log\_Prob}(w_5)$$

Final tweets polarity is positive if, total positive log probability is greater than total negative log probability else the final polarity is negative. If the none of the words are there in the word dictionary, then the final result is neutral.

In the final implementation part, the proposed framework is implemented for different case studies with different problem scenarios. Case study I concentrated on the framework implementation for the entire domain whereas Case study two concentrated for a single entity.

## **4.3 CASE STUDY**

### ***4.3.1 CASE STUDY I:***

In this case study, the proposed framework was implemented for doing the sentiment analysis for the whole automobile industry domain. In this part, tweets were collected in real time using twitter streaming on a total of 9 different automobile companies (Honda, BMW, Kia, Audi, Toyota, Lexus, Hyundai, Buick, and Mercedes Benz). The subjective words from all the tweets were extracted and final polarity score was given to each tweet.

#### **4.3.2 CASE STUDY II:**

In this case study, the proposed framework was implemented for doing the sentiment analysis for a single entity. This part used Twitter search API, and extracted all the past tweets on a single entity Regions Bank. The proposed framework was extended in this case so that it can send action emails to the assigned persons depending on the sentiment threshold.

### **4.4 SUMMARY**

This part is concentrated on the providing architectural details and implementation flow of the proposed framework. We started here by providing the architectural block diagram for the framework, followed by the process flow. In the process flow, We explained various preprocessing and data modeling steps, which were performed in the framework. In the next chapter, we will present the result section of the case studies.

## **CHAPTER – 5**

### **RESULTS AND ANALYSIS**

This section concentrates on emphasizing the results of the case studies I and II along with their analysis.

#### **5.1 CASE STUDY I**

##### ***5.1.1 EXPLORATORY ANALYSIS:***

Let's start with the Exploratory Analysis of case study I.

Total number of tweets collected = 5651.

Total number of different language tweets collected = 38

Total number of different country tweets collected = 36

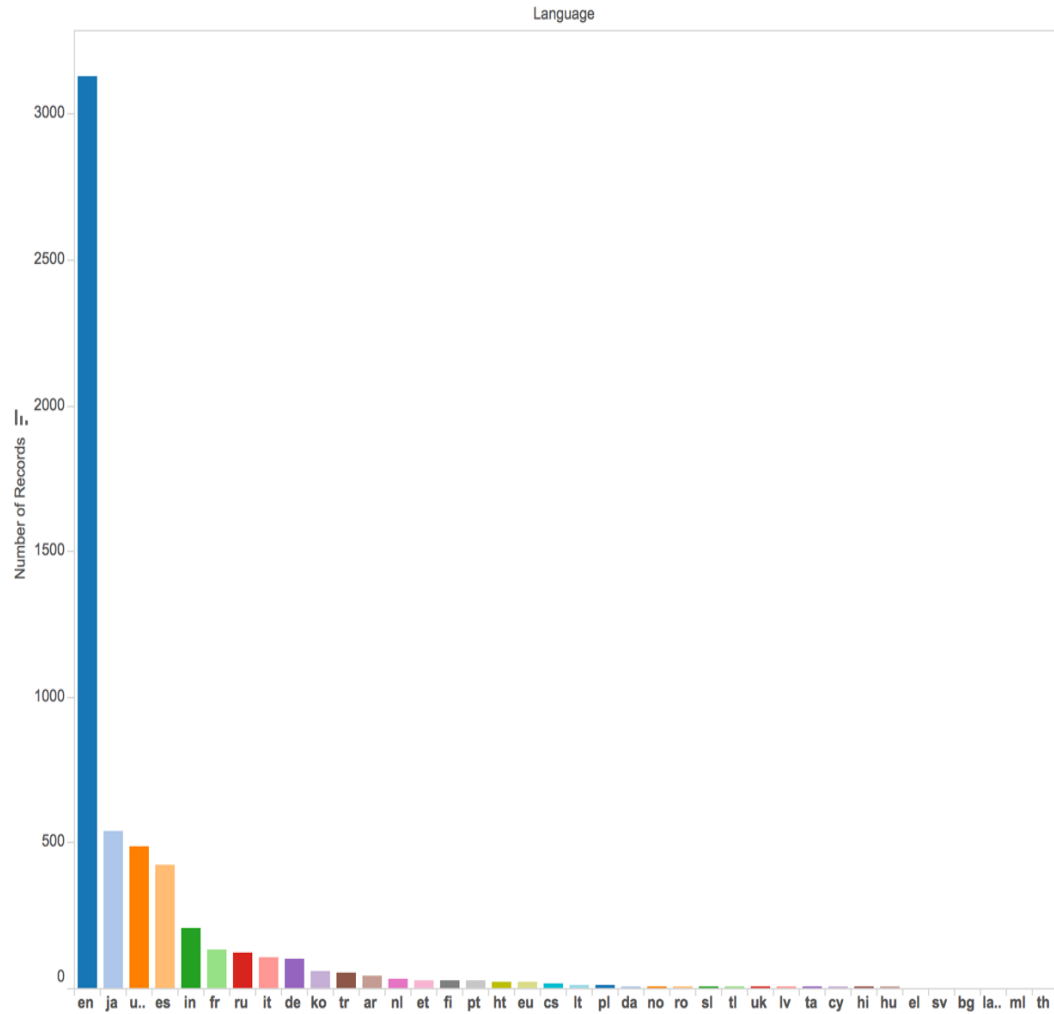


Figure 5.1: Language Distribution of Tweets

The above figure represents the language distribution of the collected tweets. Out of all the tweets 65% are in English and the remaining languages constitutes for 35% of the data. We filtered the tweets such that the final data set contains only English language tweets which accounted for a count of 3132.



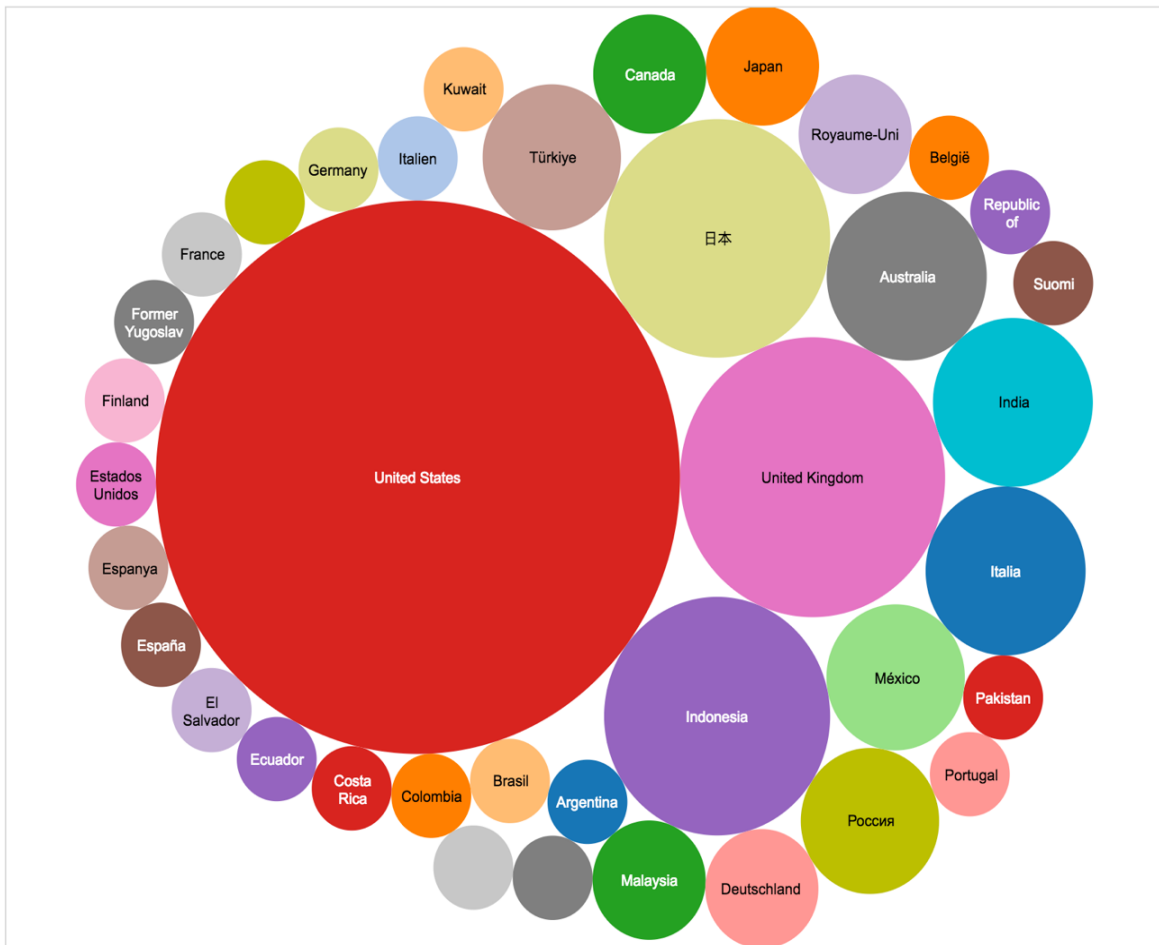


Figure 5.2: Country wide Distribution of Tweets with Count Variation

The above figure represents the count distribution of the tweets from different countries. The size of the circle represents the volume of tweets. We got the highest majority of the tweets from the United States followed by United Kingdom and a total of 36 different countries tweets were collected.

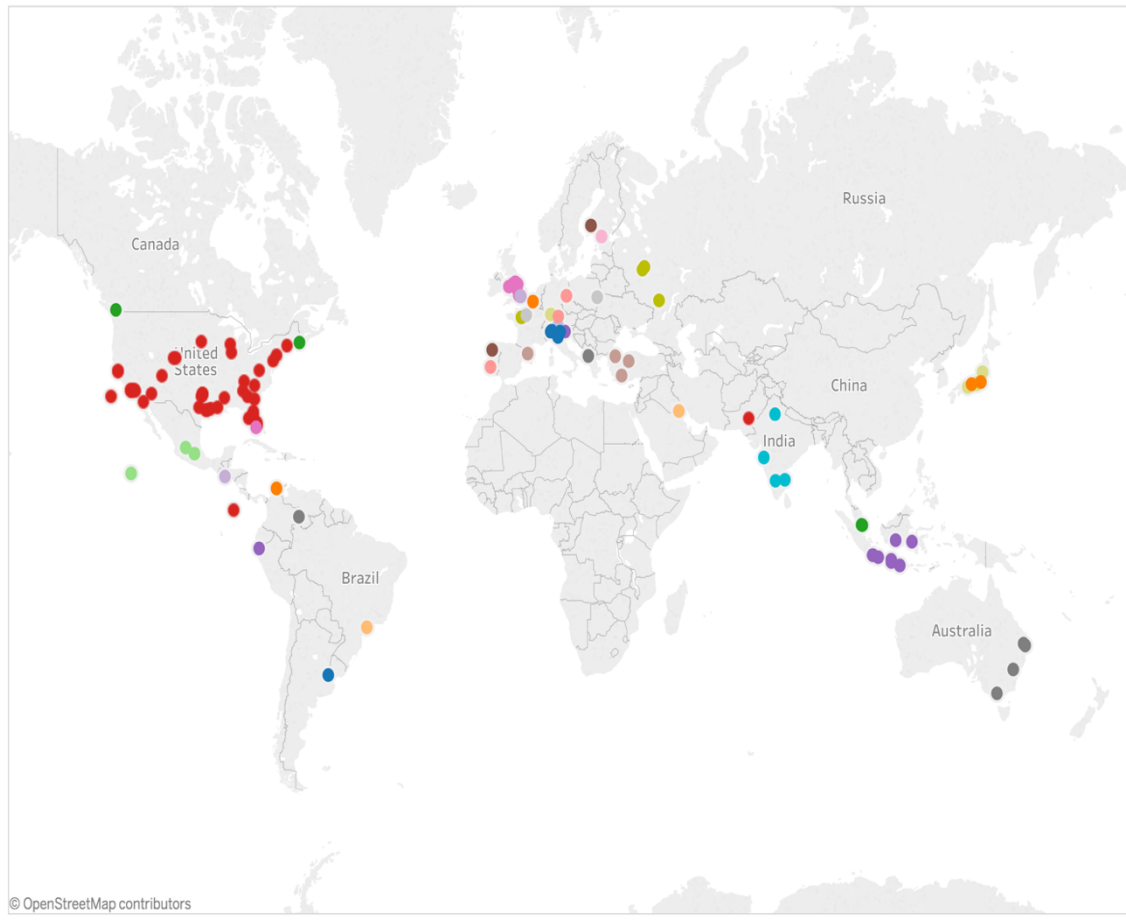


Figure 5.3: Demographic Distribution of Tweets

Figure 5.3. represents the Demographic distribution of tweets with different color represents tweets from different countries.

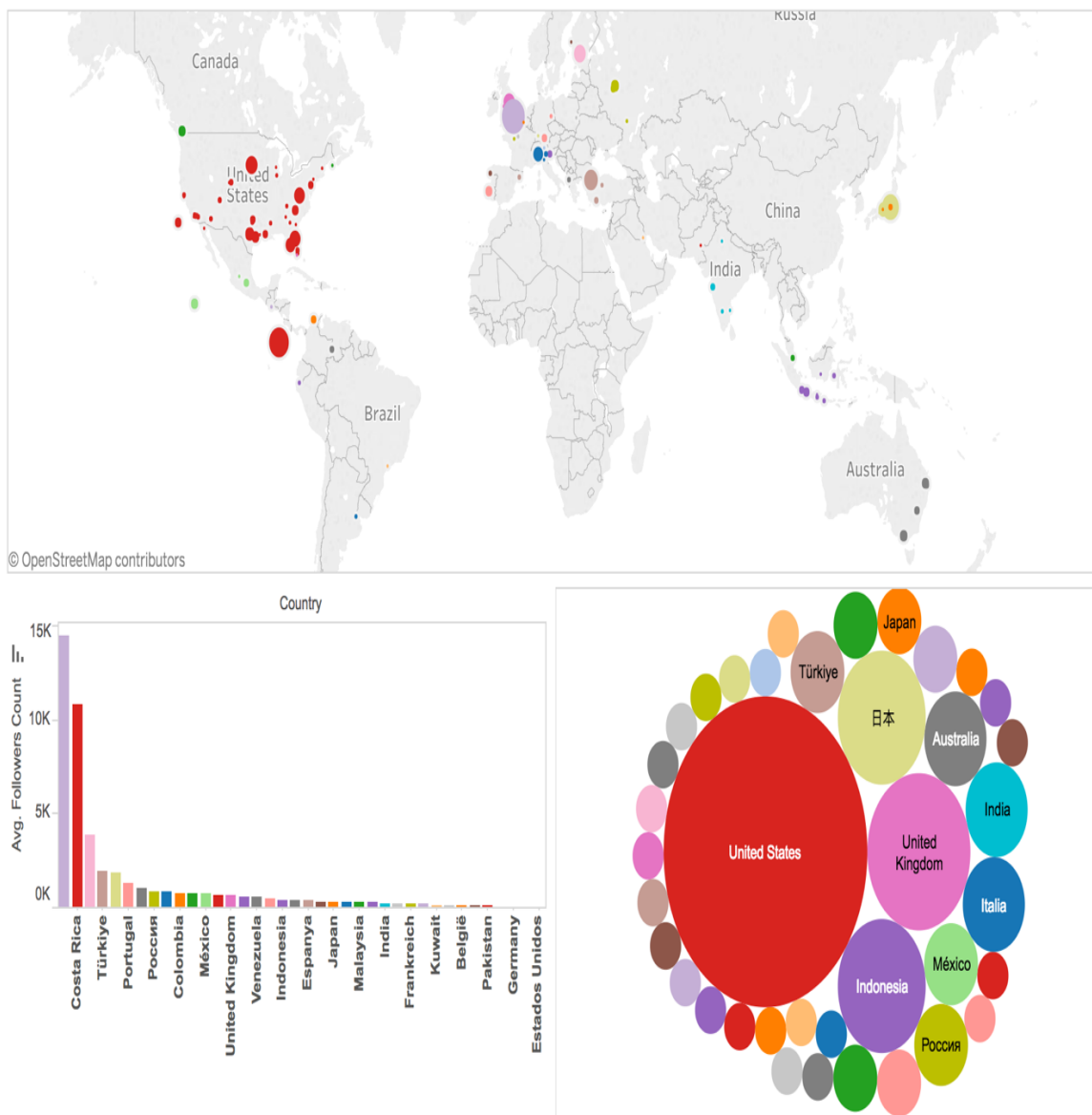


Figure 5.4: Dashboard View from Tableau

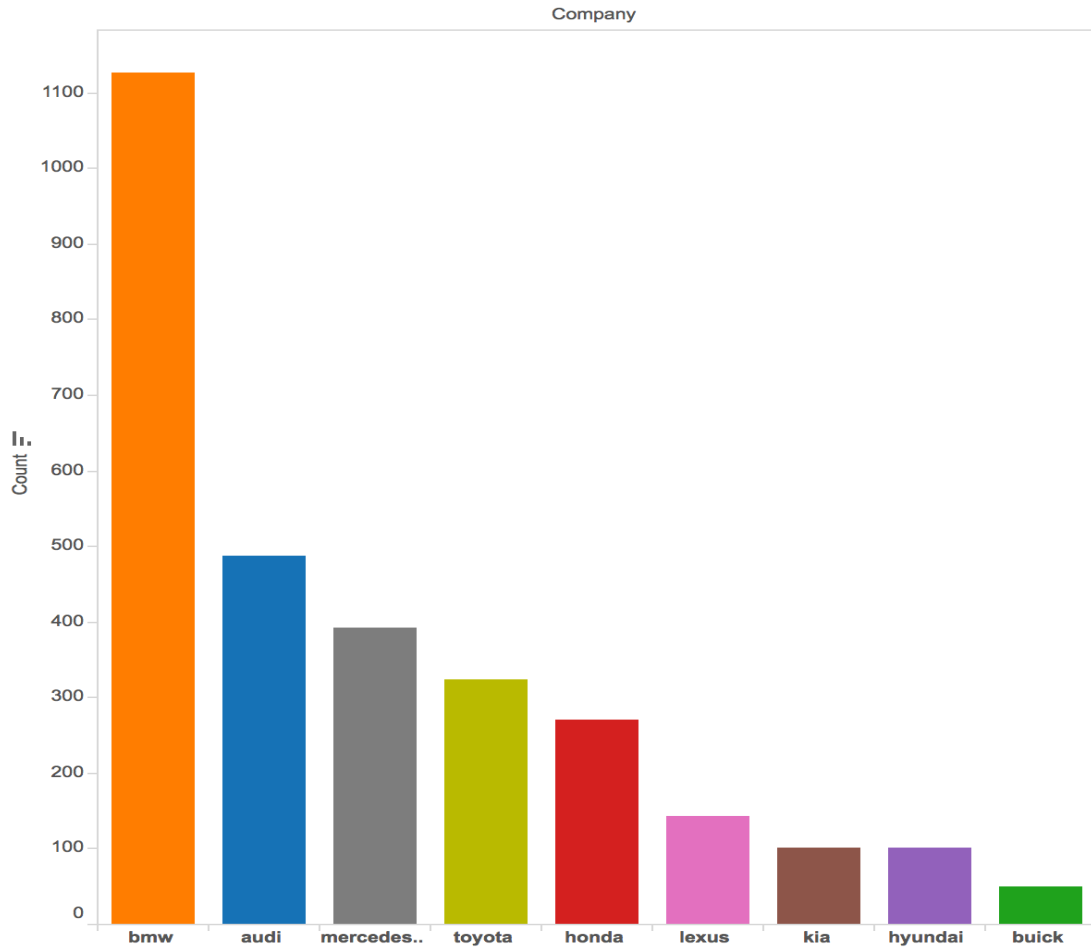


Figure 5.5: Company Distribution of Tweets

### ***5.1.2 SUMMARY OF KEY FINDINGS:***

From there graphs one can draw many inferences, However, We are mentioning only a few here.

- A total of 38 language tweets were collected and 3132 were English tweets which are almost 65% of the data collected.

- In Demographic distribution, we can see that most numbers of tweets were from the US. So US is the largest source of data when it comes to tweets of automobile industry.
- The average number of followers is more for the country Royaume-Uni, So sentiment expressed by these tweets is very important.
- BMW has got the more number of tweets with 1126 count, while Buick got the lowest number of tweets with 49 count. Therefore, we can say that BMW is the most popular one and Buick is the least popular one of all.

### **5.1.3 DATA MODELLING:**

Once the data processing and exploratory analysis part are done, the next step is to perform sentiment analysis. After performing sentiment analysis, we got 1618 positive tweets, 927 negative tweets, and 585 neutral tweets

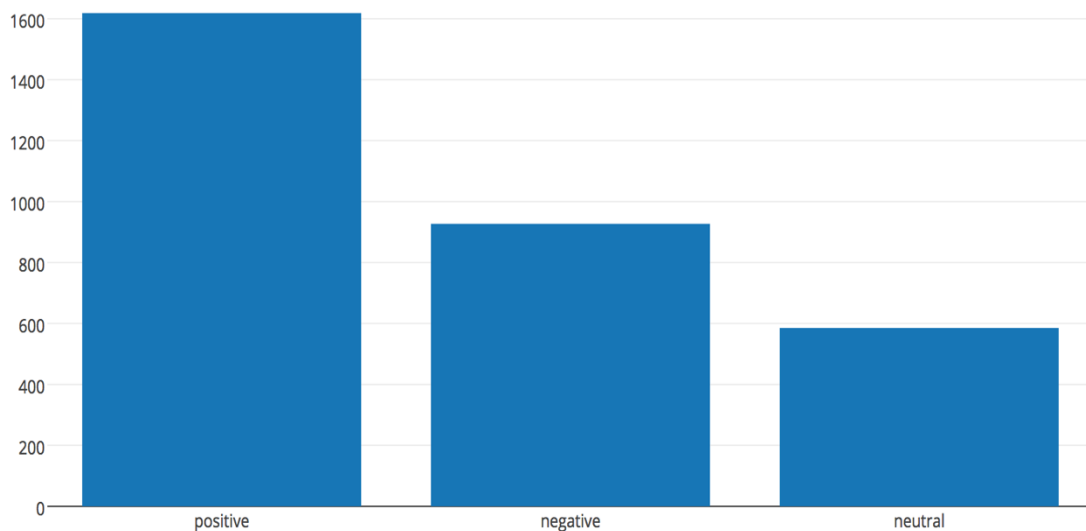


Figure 5.6: Polarity Distribution of Tweets

	text	company	tweet_words	polarity
0	RT @GuyKilfoil: The magnificent #bmw M550i XDr...	bmw	rt magnificent xdrive individual metallic s gu...	positive
1	https://t.co/AYIsnMdfeG #Carbon Fiber 2012+ #H...	honda	https tcoaylsnmdfeg carbon fiber + honda civic...	negative
2	Driving home on I-405. #bmw https://t.co/KEW8P...	bmw	home i https tcokewpvine	negative
3	Driving home on I-405. #bmw https://t.co/wEr7...	bmw	home i https tcowerfzbkr	negative
4	RT @bmwcanada: Radically more. The new #BMW #M...	bmw	new rt bmwcanada bmw m https tcosczpevxqo radi...	positive
5	RT @BMW_SA: The #BMW Concept X2 fuses a flat c...	bmw	flat robust rt bmw_sa bmw concept x coup silho...	positive
6	RT @theJDMculture: S T A N C E \n#is300\n#Lexu...	lexus	rt n lexus thejdmculture s c e theculture http...	neutral
7	RT @SAMAATV: The #BMW car that was gifted by #...	bmw	lakh rt samaatv bmw car hassannawaz maryamnawa...	negative
8	So much fun at the dc auto show #dcauto201...	audi	fun dc s auto dcauto201 moment appreciate aud...	positive
9	RT @theJDMculture: S T A N C E \n#is300\n#Lexu...	lexus	rt n lexus thejdmculture s c e theculture http...	neutral
10	Is it calling me? 🤔 #Toyota #GT86 #topgearph ...	toyota	toyota soho central private gt topgearph resid...	positive
11	RT @antonioperic: Did you know that #bmw cars ...	bmw	php rt antonioperic bmw cars https tcoauapccs...	negative
12	RT @leanettef: My babies are enjoying their 1s...	lexus	rt leanettef babies st flapanters game flapan...	positive
13	Getting there...\n#cb750 #sohc #honda #caferac...	honda	cb sohc honda caferacer flashfirecoatings http...	neutral
14	RT @sapien_jorge: I really love Audi 🤔#R8 #...	audi	rt sapien_jorge i audi r audi audi el paso htt...	positive
15	#lexus locklear pornstar most popular free por...	lexus	lexus popular free locklear pornstar porn http...	positive
16	BMW has some great Accessories. Check them out...	bmw	great accessories https tcoctbtckk bmw bmwseri...	positive
17	#Honda 1976 Honda CB HONDA CB360-T https://t.c...	honda	tcomecsthriq honda honda cb honda cbt https r...	positive
18	Der neue BMW M760Li in Palm Springs - cooles A...	bmw	mli new mli der neue bmw palm springs auto coo...	positive
19	RT @leanettef: Period 1 of the @FlaPanthers ga...	lexus	selfie lexus rt leanettef period flapanters g...	negative

Figure 5.7: Sample Output of Case Study I

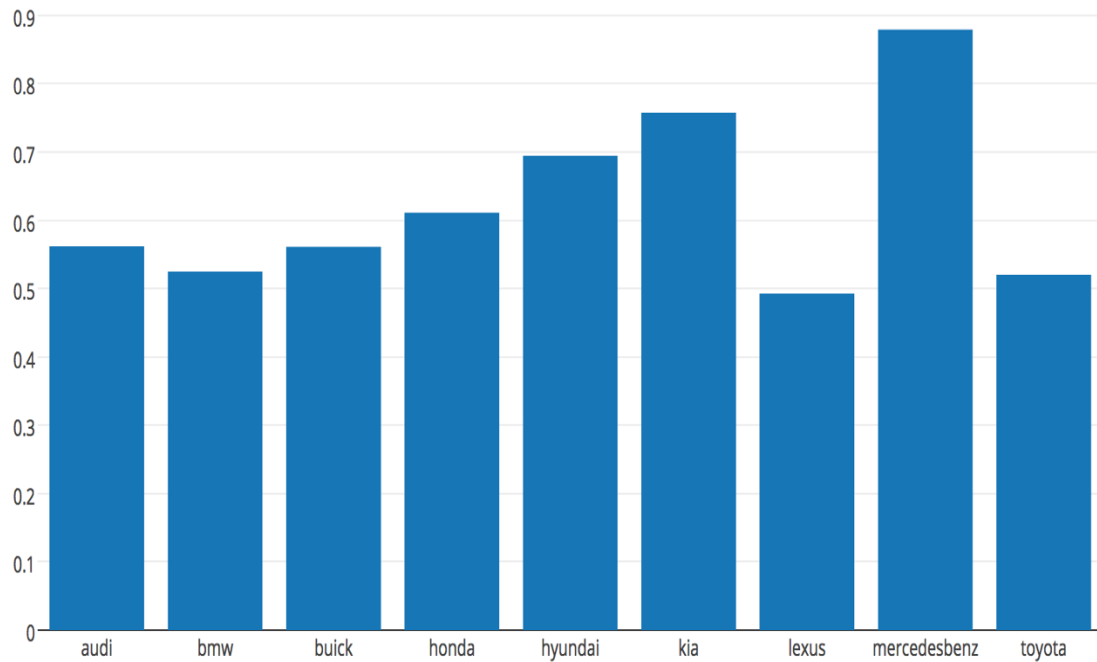


Figure 5.8: Sentiment Score of Different Automobile Companies

#### ***5.1.4 FINAL ANALYSIS:***

Of all the automobile companies chosen, Mercedes Benz has the highest sentiment score of 0.8766 and Lexus has the lowest sentiment score of 0.492. Here, we can observe that, although BMW is popular one, the final sentiment score is highest for Mercedes Benz. So popularity is different from sentiment.

## 5.2 CASE STUDY II

In Case Study II, we used the framework for performing sentiment analysis for single entity and extended with an action email system.

```
(u'askRegions that day and they also deposited 2.50 idk if it was to my account or someones else. It was on 2/10 i  
believe and i cant report\n',  
  'neg',  
  1.0),  
(u'askRegions hey someone used my card at krogers and spent 1200$ my account info had to be stolen because i was tr  
aveling to Atlanta all day\n',  
  'neg',  
  1.0)
```

Figure 5.9: Sample Output of Case Study II

Figure 5.9 represents the sentiment analysis output for single entity. Figure 5.10 represents the sample mail, which was sent by the framework to my Gmail account.

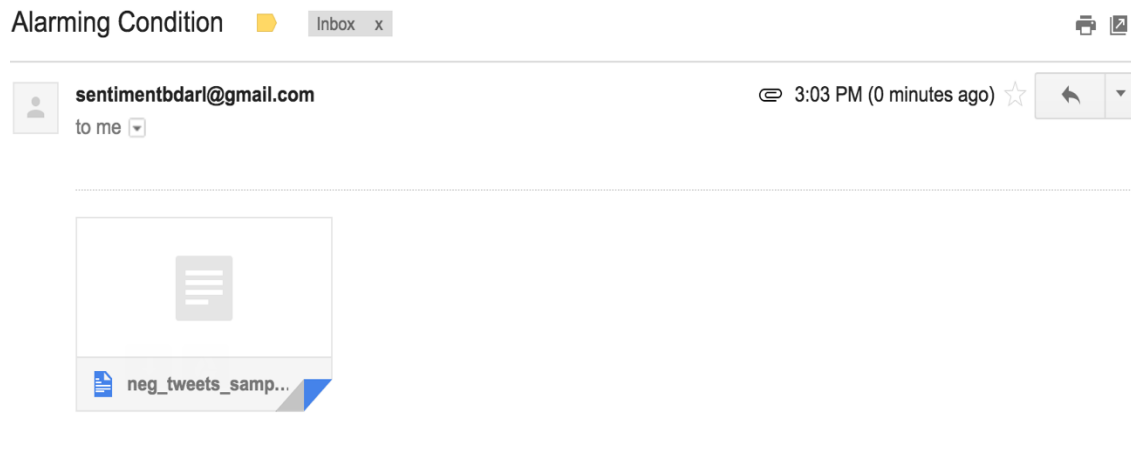


Figure 5.10: Action Email



### **5.3 SUMMARY**

This part presented the Result and Analysis sections of both the case studies, which were implemented by the proposed framework. We started by providing the exploratory analysis results and data modeling results for case study I and finally ended up by providing the results for case study II. In then next chapter, we will discuss about the future aspects of the proposed framework.

## **CHAPTER – 6**

### **CONCLUSION AND FUTURE SCOPE**

In this section, we will present the conclusion of this research along with the future scope of the developed framework.

#### **6.1 CONCLUSION**

With the proliferation of internet from commercial use to daily use, the data being generated from the social networks are increasing exponentially. Big Data platform offers various tools for accessing this data and Machine Learning makes it possible to analyze and extract the patterns from it. However, it is time-consuming and needs the expertise to write a specific application for every task. So to deal with the bottlenecks such as time and expertise we need to develop some generalized frameworks, which can be used for different tasks and can be accessible by anyone.

This whole thesis is concentrated on developing a generalized framework for social network sentiment analysis. The main aim of this framework is to make it adaptable for performing the sentiment analysis on any domain such as an automobile, banking, and telecom. This generalized framework can be used by anyone as it mitigates the underlying technical aspect of the analysis process by providing an easy to use interface. Clearly, the effectiveness of this framework on the two case studies revealed its usability to other cases.

## 6.2 FUTURE SCOPE

We believe, there is a considerable amount of work can be done in future. However, we provide some ideas here.

As of now, this framework is developed for Twitter, we can develop the framework to be suitable for other social networking sites such as Facebook and Instagram. We can also make it more appealing by creating a web console at the front end and code at the back end, where results will have automatically presented to users on the web console.

Coming to the technical perspective, for sentiment analysis, we used unigram model. However, we can use bigrams or trigrams, which guarantee more accuracy in many cases. Currently, this framework was built to support only English language, Nevertheless, we can improve the framework to support multi-languages [35]. Although, we hard coded some visualization plots, we can make these things automated by purchasing some third party libraries.

Just as to provide some beam of light for future research, in case study II, we expanded the proposed framework by appending an action email system. So we are confident enough to say that there are significant ways left for future scope and opportunities of this framework can be taken to any further level.

## REFERENCES:

- [1] Andreas M Kaplan and Michael Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business horizons*, 53(1):59–68, 2010.
- [2] Search Engine Journal: <https://www.searchenginejournal.com/growth-social-media-v-3-0-infographic/155115>
- [3] Twitter: <https://dev.twitter.com/overview/api>
- [4] R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. A comprehensive grammar of the English language. Longman, 1985.
- [5] J. M. Wiebe. Tracking point of view in narrative. *Computational Linguistics*, 20:233–287, 1994.
- [6] Klaus R Scherer. Vocal communication of emotion: A review of research paradigms. *Speech communication*, 40(1):227–256, 2003.
- [7] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. In *The Semantic Web–ISWC 2012*, pages 508–524. Springer, 2012.
- [8] Nadia FF da Silva, Eduardo R Hruschka, and Estevam R Hruschka. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, 66:170–179, 2014.
- [9] John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, pages 440–447, 2007.
- [10] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.
- [11] Bishan Yang and Claire Cardie. Joint inference for fine-grained opinion extraction. In *ACL (1)*, pages 1640–1649, 2013.
- [12] Bing Liu. Sentiment analysis and subjectivity. *Handbook of natural language processing- ing*, 2:627–666, 2010.

- [13] Samuel, Arthur L. (1959). "Some studies in machine learning using the game of checkers". IBM Journal of research and development.
- [14] <http://www.britannica.com/EBchecked/topic/1116194/machine-learning>
- [15] Bishop, C. M. (2006), Pattern Recognition and Machine Learning, Springer, ISBN 0-387-31073-8
- [16] Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar (2012) Foundations of Machine Learning, MIT Press ISBN 978-0-262-01825-8.
- [17] Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955.
- [18] Steve Lohr (1 February 2013). "The Origins of 'Big Data': An Etymological Detective Story". New York Times. Retrieved 28 September 2016.
- [19] Laney, Douglas. "3D Data Management: Controlling Data Volume, Velocity and Variety" (PDF). Gartner. Retrieved 6 February 2001.
- [20] Research Access: <http://researchaccess.com/2014/08/the-fourth-v-of-big-data>
- [21] <http://www.ey.com/gl/en/services/advisory/ey-big-data-big-opportunities-big-challenges>
- [22] <http://www.claimsjournal.com/news/national/2013/07/29/233805.html>
- [23] "Welcome to Apache Hadoop!". [hadoop.apache.org](http://hadoop.apache.org). Retrieved 2016-08-25.
- [24] <http://ercoppa.github.io/HadoopInternals/HadoopArchitectureOverview.html>
- [25] Hive: <http://hive.apache.org>
- [26] Continuum Analytics: <https://docs.continuum.io>
- [27] Anaconda Navigator: <https://docs.continuum.io/anaconda/navigator>
- [28] Conda: <https://conda.io/docs/index.html>
- [29] Jupyter Notebook: <http://jupyter.org>
- [30] <http://jupyter-notebook.readthedocs.io/en/latest/examples/Notebook>
- [31] <http://spark.apache.org/docs/latest/streaming-programming-guide.html>
- [32] <http://spark.apache.org/docs/latest/graphx-programming-guide.html>

- [33] <https://databricks.com/blog/2016/07/14/a-tale-of-three-apache-spark-apis-rdds-dataframes-and-datasets.html>
- [34] Spark: <http://spark.apache.org>
- [35] S. Narr, M. Hifulfenhaus, and S. Albayrak, Language-independent twitter sentiment analysis.”, The 5th SNA-KDD Workshop, 2011.
- [36] Persuasiveness in Social Multimedia: The Role of Communication Modality and the Challenge of Crowdsourcing Annotations, University of South California, [itc.usc.edu](http://itc.usc.edu)
- [37] Weimiao Feng, Jianguo Sun, Liguozhang, Cuiling Cao and Qing Yang, A support vector machine based naive Bayes algorithm for spam filtering, Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International
- [38] <http://machinelearningmastery.com/parametric-and-nonparametric-machine-learning-algorithms>
- [39] [https://sebastianraschka.com/faq/docs/parametric\\_vs\\_nonparametric.html](https://sebastianraschka.com/faq/docs/parametric_vs_nonparametric.html)

**APPENDIX**  
**PROGRAM CODE**

Below program code is the copy version of .py code generated from the Jupyter Notebooks. Appropriate sections were created in the program code for detailed understanding

## CASE STUDY I

### CODE:

```
# coding: utf-8
```

### Twitter Streaming

```
# Importing Libraries
from tweepy.streaming import StreamListener
from tweepy import OAuthHandler
from tweepy import Stream

# Twitter Credentials
CONSUMER_KEY = ""
CONSUMER_SECRET = ""
OAUTH_TOKEN = ""
OAUTH_TOKEN_SECRET = ""
# Please include your credentials in between the quotes

# Listener
class Stream_Listen(StreamListener):

    def on_data(self, data):
        file_tweets = open("tweets.csv","a")
        file_tweets.write(data)
        file_tweets.write("\n")
        file_tweets.close()
        print data
```



```

        return True
    def on_error(self, status):
        print status
        time.sleep(5)
if __name__ == '__main__':

    # For Authentication
    auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
    auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
    twitter_stream = Stream(auth, Stream_Listen())

    # Filter to get the tweets on the certain intervals
    Automobiles =
    ['Honda','BMW','Kia','Audi','Toyota','Lexus','Hyundai','Buick','MercedesBenz']

    # twitter_stream.filter(track =
    ['#Honda','#BMW','#Kia','#Audi','#Toyota','#Lexus','#Hyundai','#Buick','#MercedesBenz',
    '#Honda','#BMW','#Kia','#Audi','#Toyota','#Lexus','#Hyundai','#Buick','#MercedesBenz'],
    async=True)

```

### **Exploratory Analysis**

```

# Importing Necessary Libraries
import pandas as pd
import json
import matplotlib.pyplot as plt
import re
import os

# Reading Tweets.csv file
tweets = []

```

```
with open("tweets.csv","r") as f:
```

```
    for line in f:
```

```
        try:
```

```
            tweet = json.loads(line)
```

```
            tweets.append(tweet)
```

```
        except:
```

```
            continue
```

```
# Total Tweets
```

```
len(tweets)
```

```
# Generating a Data Frame
```

```
#tweet_fields = [u'contributors', u'coordinates', u'created_at', u'entities',
```

```
#             u'favorite_count', u'favorited', u'filter_level', u'geo', u'id',
```

```
#             u'id_str', u'in_reply_to_screen_name', u'in_reply_to_status_id',
```

```
#             u'in_reply_to_status_id_str', u'in_reply_to_user_id',
```

```
u'in_reply_to_user_id_str',
```

```
#             u'lang', u'place', u'retweet_count', u'retweeted', u'source',
```

```
#             u'text', u'timestamp_ms', u'truncated', u'user']
```

```
tweets_frame = pd.DataFrame()
```

```
tweets_frame['contributors'] = map(lambda tweet: tweet['contributors'] if  
tweet['contributors'] != None else None, tweets )
```

```
tweets_frame['coordinates'] = map(lambda tweet: tweet['coordinates'] if  
tweet['coordinates'] != None else None, tweets )
```

```
tweets_frame['created_at'] = map(lambda tweet: tweet['created_at'] if tweet['created_at']  
!= None else None, tweets )
```

```
tweets_frame['entities'] = map(lambda tweet: tweet['entities'] if tweet['entities'] != None  
else None, tweets )
```

```
tweets_frame['favorite_count'] = map(lambda tweet: tweet['favorite_count'] if  
tweet['favorite_count'] != None else None, tweets )
```

```

tweets_frame['favorited'] = map(lambda tweet: tweet['favorited'] if tweet['favorited'] !=
None else None, tweets )
tweets_frame['filter_level'] = map(lambda tweet: tweet['filter_level'] if
tweet['filter_level'] != None else None, tweets )
tweets_frame['geo'] = map(lambda tweet: tweet['geo'] if tweet['geo'] != None else None,
tweets )
tweets_frame['id'] = map(lambda tweet: tweet['id'] if tweet['id'] != None else None,
tweets )
tweets_frame['id_str'] = map(lambda tweet: tweet['id_str'] if tweet["id_str"] != None else
None, tweets )
tweets_frame['in_reply_to_screen_name'] = map(lambda tweet:
tweet['in_reply_to_screen_name'] if tweet['in_reply_to_screen_name'] != None else
None, tweets )
tweets_frame['in_reply_to_status_id'] = map(lambda tweet: tweet['in_reply_to_status_id']
if tweet['in_reply_to_status_id'] != None else None, tweets )
tweets_frame['in_reply_to_status_id_str'] = map(lambda tweet:
tweet['in_reply_to_status_id_str'] if tweet['in_reply_to_status_id_str'] != None else None,
tweets )
tweets_frame['in_reply_to_user_id'] = map(lambda tweet: tweet['in_reply_to_user_id'] if
tweet['in_reply_to_user_id'] != None else None, tweets )
tweets_frame['in_reply_to_user_id_str'] = map(lambda tweet:
tweet['in_reply_to_user_id_str'] if tweet['in_reply_to_user_id_str'] != None else None,
tweets )
tweets_frame['lang'] = map(lambda tweet: tweet['lang'] if tweet['lang'] != None else
None, tweets )
tweets_frame['place'] = map(lambda tweet: tweet['place'] if tweet['place'] != None else
None, tweets )
tweets_frame['retweet_count'] = map(lambda tweet: tweet['retweet_count'] if
tweet['retweet_count'] != None else None, tweets )
tweets_frame['retweeted'] = map(lambda tweet: tweet['retweeted'] if tweet['retweeted'] !=
None else None, tweets )

```

```

tweets_frame['source'] = map(lambda tweet: tweet['source'] if tweet['source'] != None
else None, tweets )
tweets_frame['text'] = map(lambda tweet: tweet['text'] if tweet['text'] != None else None,
tweets )
tweets_frame['timestamp_ms'] = map(lambda tweet: tweet['timestamp_ms'] if
tweet['timestamp_ms'] != None else None, tweets )
tweets_frame['truncated'] = map(lambda tweet: tweet['truncated'] if tweet['truncated'] !=
None else None, tweets )
tweets_frame['user'] = map(lambda tweet: tweet['user'] if tweet['user'] != None else None,
tweets )
tweets_frame['country'] = map(lambda tweet: tweet['place']['country'] if tweet['place'] !=
None else "None", tweets )
tweets_frame['followers_count'] = map(lambda tweet: tweet['user']['followers_count'] if
tweet['user'] != None else None, tweets )
tweets_frame['full_name'] = map(lambda tweet: tweet['place']['full_name'] if
tweet['place'] != None else None, tweets )
tweets_frame['Longitude'] = map(lambda tweet:
tweet['place']['bounding_box']['coordinates'][0][0][0] if tweet['place'] != None else None,
tweets )
tweets_frame['Latitude'] = map(lambda tweet:
tweet['place']['bounding_box']['coordinates'][0][0][1] if tweet['place'] != None else None,
tweets )

# Getting Language Distribution Tweets
tweets_frame[['lang']].to_excel("tableauLang.xlsx")
Frame_Locations_on = tweets_frame[tweets_frame.country != "None"]

# Tweets with location setting on
len(Frame_Locations_on)

# Getting unique tweets

```

```

Frame_Locations_on_unique = Frame_Locations_on.drop_duplicates('full_name')
Tableau_Frame =
Frame_Locations_on_unique[['country','full_name','Latitude','Longitude','followers_count']]

# Saving Tableau Frame for Tableau views
Tableau_Frame.to_excel("Tableau.xlsx")

# Function for Comparison
def word_occurance(word, text):
    word = word.lower()
    text = text.lower()
    match = re.search(word, text)
    if match:
        return True
    return False

# Before moving further lets get the tweets which are in english language
tweets_frame_english = tweets_frame[tweets_frame['lang']=="en"]

# Automobile Industries
automobiles = ['honda','bmw','kia','audi','toyota','lexus','hyundai','buick','mercedesbenz']

# Getting the no.of tweets for each automobile company
counts = {}
counts['honda'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda tweet: word_occurance('honda', tweet))])
counts['bmw'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda tweet: word_occurance('bmw', tweet))])
counts['kia'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda tweet: word_occurance('kia', tweet))])

```

```

counts['audi'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda
tweet: word_occurance('audi', tweet))])
counts['toyota'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda
tweet: word_occurance('toyota', tweet))])
counts['lexus'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda
tweet: word_occurance('lexus', tweet))])
counts['hyundai'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda
tweet: word_occurance('hyundai', tweet))])
counts['buick'] = len(tweets_frame_english[tweets_frame_english['text'].apply(lambda
tweet: word_occurance('buick', tweet))])
counts['mercedesbenz'] =
len(tweets_frame_english[tweets_frame_english['text'].apply(lambda tweet:
word_occurance('mercedesbenz', tweet))])

```

```

# Getting automobile company with highest and lowest tweets
print "highest Tweets company: {} "
.format(counts.keys()[counts.values().index(max(counts.values()))])
print "highest Value = {}" .format(max(counts.values()))
print "lowest Tweets company: {} "
.format(counts.keys()[counts.values().index(min(counts.values()))])
print "lowest Value = {}" .format(min(counts.values()))

```

```

# For visualization Purpose
counts_frame = pd.DataFrame(counts.values(),counts.keys(),columns=['count'])
counts_frame.to_excel("frame.xlsx")
counts_frame

```

## **Data Modelling**

```
# Function For assigning particular company for each tweet
def airlines_name(tweet, automobiles =
['honda','bmw','kia','audi','toyota','lexus','hyundai','buick','mercedesbenz']):
    tweet = tweet.lower()
    for x in automobiles:
        match = re.search(x.lower(),tweet)
        if match:
            return x
    return None

# Adding new column as company
tweets_frame_english['Company'] = tweets_frame_english['text'].apply(lambda x:
airlines_name(x))
tweets_frame_english.groupby('Company').Company.count()
tweets_frame_english.head()

# Using PySpark
print sc
print sqlContext

# Language Processing module
# Importing necessary Libraries
import nltk
import numpy as np
from nltk.tokenize import word_tokenize, sent_tokenize
from nltk.corpus import stopwords
from sklearn.feature_extraction import text
from pyspark.sql.functions import udf
from pyspark.sql.functions import split
```

```

def required_words(tweet):
    stop_words_nltk = stopwords.words('english')
    stop_words_sklearn = text.ENGLISH_STOP_WORDS
    stop = set(stop_words_nltk) | set(stop_words_sklearn)
    punc = set(["#", ".", "{", "}", " ", "@", "\\", "/", " ", "!", "?", " ", ""])
    words = [re.sub(r"^\x00-\x7F", "", re.sub(r"[-\d\.\?]", "", x.lower())) for x in
word_tokenize(tweet) if x not in stop | punc]
    words_replaced = filter(lambda a: a != "", words)
    pos_words = nltk.pos_tag(words_replaced)
    #print pos_words
    adjectives = [x[0] for x in pos_words if x[1] in ["JJ", "JJR", "JJS"]]
    nouns = [x[0] for x in pos_words if x[1] in ["NN", "NNP", "NNPS", "NNS"]]
    verbs = [x[0] for x in pos_words if x[1] in ["RBS", "RBR", "RB", "VB"]]
    #return adjectives+nouns
    return " ".join(adjectives+nouns+verbs)

spark_func = udf(required_words)

Spark_Data_Frame =
sqlContext.createDataFrame(tweets_frame_english[['text', 'Company', 'followers_count']])
tweets_spark_frame =
Spark_Data_Frame.select('text', 'Company', spark_func(Spark_Data_Frame['text']).alias('t
weet_words'))
tweets_spark_frame.show()

# Function For taking sentiment words with log probabilities
def sentiment_split(file_name):
    with open(file_name, "r") as f:
        happy_prob_log = {}
        sad_prob_log = {}
        for line in f:
            try:
                tokenize = line.split(",")

```



```

        happy_prob_log[tokenize[0]] = float(tokenize[1])
        sad_prob_log[tokenize[0]] = float(tokenize[2])
    except:
        continue
    return happy_prob_log, sad_prob_log

happy_probs, sad_probs = sentiment_split("Senti_words.csv")

# Function For Naive Bayes Classifier
def classifier_sentiment(tweets, happy_probs = happy_probs, sad_probs = sad_probs):
    #c = c+1
    #if c == 781:
    #return 0
    tweet = tweets.split(" ")
    probs_positive = [happy_probs[word] for word in tweet if word in
happy_probs.keys()]
    #print probs_positive
    probs_negative = [sad_probs[word] for word in tweet if word in sad_probs.keys()]
    #print probs_negative
    Total_log_pos_prob = np.sum(probs_positive)
    print Total_log_pos_prob
    #if Total_log_pos_prob == -20.73131:
    #print "gotyoupos"
    Total_log_neg_prob = np.sum(probs_negative)
    #if Total_log_neg_prob == -20.73131:
    #print "gotyou"
    print Total_log_neg_prob
    if Total_log_pos_prob > Total_log_neg_prob:
        return "positive"
    elif Total_log_pos_prob == Total_log_neg_prob:
        return "neutral"

```

```

else:
    return "negative"

spark_bayes_func = udf(classifier_sentiment)
Final_Spark_Frame =
tweets_spark_frame.select('text','company','tweet_words',spark_bayes_func(tweets_spark
_frame.tweet_words).alias('polarity'))
Final_Spark_Frame.show()
Final_Spark_Frame.toPandas()
Final_Pandas_Frame = Final_Spark_Frame.toPandas()
Final_Pandas_Frame.head(20)
Final_Pandas_Frame.groupby('polarity').polarity.count()
scores = []
for x in list(Final_Pandas_Frame.polarity):
    if x == "positive":
        scores.append(1)
    if x == "negative":
        scores.append(0)
    if x == "neutral":
        scores.append(0.5)
Final_Pandas_Frame['scores'] = scores
Average_Scores = Final_Pandas_Frame.groupby('company').scores.mean()
Average_Scores.keys()
Final_Pandas_Frame['followers_count'] = list(tweets_frame_english.followers_count)
Final_Pandas_Frame
Final_Pandas_Frame.to_excel("Tableau_Frame.xlsx")#, encoding="UTF-8")

```

**Skill Set Used:**

Topics: Descriptive Statistics, Machine Learning, Natural Language Processing,  
Sentiment analysis

Machine Learning Algorithms: Naive Bayes algorithm

Natural Language Processing: Parsing, chunking, Parts of Speech tagging, Named Entity  
Recognition

Compute: Python, PySpark

Data Tools: Jupyter Notebook, AWS EC2 Instance

Data Visualization: Plotly, Tableau, matplotlib

Version Control: Git

**CASE STUDY II****CODE:**

# coding: utf-8

**Sentiment Module Code**

```
# Importing libraries
import nltk
import random
from nltk.classify.scikitlearn import SklearnClassifier
import pickle
from sklearn.naive_bayes import MultinomialNB, BernoulliNB
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.svm import SVC, LinearSVC, NuSVC
from nltk.classify import ClassifierI
from nltk.tokenize import word_tokenize
from scipy.stats import mode
from collections import Counter
```

```

# Voting best classifier
class VoteClassifier(ClassifierI):
    def __init__(self, *classifiers):
        self._classifiers = classifiers
    def classify(self, features):
        votes = []
        for c in self._classifiers:
            v = c.classify(features)
            votes.append(v)
        return Counter(votes).most_common(1)[0][0]
    def confidence(self, features):
        votes = []
        for c in self._classifiers:
            v = c.classify(features)
            votes.append(v)
        choice_votes = votes.count(Counter(votes).most_common(1)[0][0])
        conf = float(choice_votes) / float(len(votes))
        return conf

import sys
reload(sys)
sys.setdefaultencoding('utf8')
req_tags = ["JJ", "JJR", "JJS"]
documents = []
all_words = []
with open("positive.txt", "r") as f:
    positive_words = []
    for line in f:
        try:
            documents.append( (line, "pos") )
            pos_tag = nltk.pos_tag(tokenize)
            positive_words.append([x[0].lower() for x in pos_tag if x[1] in req_tags])

```

```

        except:
            continue

with open("negative.txt","r") as f:
    negative_words = []
    for line in f:
        try:
            documents.append( (line, "neg") )
            tokenize = word_tokenize(line)
            neg_tag = nltk.pos_tag(tokenize)
            negative_words.append([x[0].lower() for x in neg_tag if x[1] in req_tags])
        except:
            continue

positive_words = [x for y in positive_words for x in y]
negative_words = [x for y in negative_words for x in y]
freq_pos = nltk.FreqDist(positive_words)
prop_pos = [ {freq_pos.keys()[x]:float(freq_pos.values()[x]/13431.00)} for x in
range(len(freq_pos))]
freq_neg = nltk.FreqDist(negative_words)
prop_neg = [ {freq_neg.keys()[x]:float(freq_neg.values()[x]/13431.00)} for x in
range(len(freq_neg))]
features_words = nltk.FreqDist(positive_words+negative_words).keys()
document_save = open("documents.pickle","wb")
pickle.dump(documents, document_save)
document_save.close()
features_save = open("features.pickle","wb")
pickle.dump("features_words", features_save)
features_save.close()

def find_features(document):

```

```

words = word_tokenize(document)
features = {}
for w in features_words:
    features[w] = (w in words)
return features
featuresets = [(find_features(rev), category) for (rev, category) in documents]
random.shuffle(featuresets)
len(featuresets)

training_data = featuresets[:9000]
testing_data = featuresets[9000:]

# Classifiers
base_classifier = nltk.NaiveBayesClassifier.train(training_data)
MNB_classifier = SklearnClassifier(MultinomialNB()).train(training_data)
BernoulliNB_classifier = SklearnClassifier(BernoulliNB()).train(training_data)
LinearSVC_classifier = SklearnClassifier(LinearSVC()).train(training_data)
LogisticRegression_classifier =
SklearnClassifier(LogisticRegression()).train(training_data)

save_classifier = open("algorithms_pickle/naive_bayes.pickle", "wb")
pickle.dump(base_classifier, save_classifier)
save_classifier.close()

save_classifier = open("algorithms_pickle/MNB.pickle", "wb")
pickle.dump(MNB_classifier, save_classifier)
save_classifier.close()

save_classifier = open("algorithms_pickle/Berno.pickle", "wb")
pickle.dump(BernoulliNB_classifier, save_classifier)
save_classifier.close()

```

```
save_classifier = open("algorithms_pickle/LineaSVC.pickle","wb")
pickle.dump(LinearSVC_classifier, save_classifier)
save_classifier.close()
```

```
save_classifier = open("algorithms_pickle/Logistic.pickle","wb")
pickle.dump(LogisticRegression_classifier, save_classifier)
save_classifier.close()
```

```
open_file = open("algorithms_pickle/naive_bayes.pickle","rb")
Naive_Bayes_Classifier = pickle.load(open_file)
open_file.close()
```

```
open_file = open("algorithms_pickle/MNB.pickle","rb")
MNB_Classifier = pickle.load(open_file)
open_file.close()
```

```
open_file = open("algorithms_pickle/Berno.pickle","rb")
Berno_Classifier = pickle.load(open_file)
open_file.close()
```

```
open_file = open("algorithms_pickle/LineaSVC.pickle","rb")
Linear_SVC_Classifier = pickle.load(open_file)
open_file.close()
```

```
open_file = open("algorithms_pickle/Logistic.pickle","rb")
Logistic_Classifier = pickle.load(open_file)
open_file.close()
```

```
Classifier_winner = VoteClassifier(Naive_Bayes_Classifier,
                                   MNB_Classifier,
                                   Berno_Classifier,
```

```
Linear_SVC_Classifier,  
Logistic_Classifier)
```

```
def sentiment(tweet):  
    tweet_words = find_features(tweet)  
    return  
Classifier_winner.classify(tweet_words),Classifier_winner.confidence(tweet_words)
```

```
sentiment("This movie was awesome! The acting was great, plot was wonderful, and  
there were pythons....so yea!")
```

### **Implementation for Regions Bank**

```
# Importing libraries  
from tweepy.streaming import StreamListener  
from tweepy import OAuthHandler  
from tweepy import Stream  
import tweepy  
import socket  
import json  
import pandas as pd  
import Sentiment_Module as senti  
import codecs  
import smtplib  
from email.mime.text import MIMEText  
from email.mime.multipart import MIMEMultipart  
from email.MIMEBase import MIMEBase  
from email import Encoders  
  
# Twitter Credentials  
CONSUMER_KEY = ""
```



```

CONSUMER_SECRET = ""
OAUTH_TOKEN = ""
OAUTH_TOKEN_SECRET = ""

# Twitter Authentication
auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)

# Api Initiation
api = tweepy.API(auth)

# Creating a Data Frame For Storing Tweets
Tweets_Frame = pd.DataFrame()

# For getting tweets on askRegions tag
for tweet in tweepy.Cursor(api.search,q='askRegions').items():
    print tweet.text
    msg = tweet.text.encode('UTF-8')
    file_tweets = open("tweets_sentiment_final.csv","a")
    if senti.sentiment(tweet.text):
        file_tweets.write(tweet.text)
        file_tweets.write("\n")
        file_tweets.close()
    else:
        continue
    print msg
    senti.sentiment(tweet.text)

with codecs.open("tweets_sentiment_final.csv","r","utf-8") as f:
    tweets_polarity = []
    for line in f:

```

```
tweets_polarity.append((line,senti.sentiment(line)[0],senti.sentiment(line)[1]))
```

```
DataFrame = pd.DataFrame(tweets_polarity,columns=["tweets","polarity","vote_ratio"])
```

### **Action Email Implementation**

```
counts = DataFrame.groupby('polarity').polarity.count()
if counts[0]>counts[1]: # If negative tweets are greater than positive tweets
    # Connecting to the google server
    msg = MIMEMultipart()
    msg['From'] = 'sentimentbdar@gmail.com'
    msg['To'] = 'kbsriharsha@gmail.com'
    msg['Subject'] = "Alarming Condition"
    part = MIMEBase('application', "octet-stream")
    part.set_payload(open("neg_tweets_sample.txt", "rb").read())
    Encoders.encode_base64(part)
    part.add_header('Content-Disposition', 'attachment;
filename="neg_tweets_sample.txt")
    msg.attach(part)
    mail_server = smtplib.SMTP("smtp.gmail.com:587")

    #Identifying ourselves
    mail_server.ehlo
    mail_server.starttls()

    #Logging in with my credentials
    mail_server.login("*****@gmail.com", "*****")
    mail_server.sendmail("*****@gmail.com", "*****88@gmail.com",msg.as_
string())
    mail_server.close()
```

## **Skill Set Used**

Topics: Machine Learning, Natural Language Processing, Sentiment analysis,  
Networking

Machine Learning Algorithms: Naive Bayes algorithm, Multinomial Naive Bayes,  
Bernoulli Naive Bayes, Logistic Regression, Linear SVC

Networking: Basic Protocols (SMTP, TLS), Server-connection-identification </p>

Robustness: Ranking method

Natural Language Processing: Parsing, chunking, Parts of Speech tagging, Named Entity  
Recognition

Compute: python

Data Tools: Jupyter Notebook, Spyder

Version Control: Git