



KADIR HAS UNIVERSITY  
SCHOOL OF GRADUATE STUDIES  
DEPARTMENT OF ENGINEERING AND NATURAL SCIENCES

# **TWITTER SENTIMENT ANALYSIS VIA MACHINE LEARNING**

KEMAL MAHMUT KAŞGARLI  
ASSOC. PROF. DR. TANER ARSAN

MASTER'S DEGREE THESIS


ISTANBUL, FEBRUARY 2021

KEMAL MAHMUT KAŞGARLI

Master's Degree Thesis

2021

# **TWITTER SENTIMENT ANALYSIS VIA MACHINE LEARNING**



KEMAL MAHMUT KAŞGARLI  
ASSOC. PROF. DR. TANER ARSAN

MASTER'S DEGREE THESIS

SUBMITTED TO THE SCHOOL OF GRADUATE STUDIES  
WITH THE AIM TO MEET THE PARTIAL REQUIREMENTS REQUIRED TO  
RECEIVE A MASTER'S DEGREE IN THE DEPARTMENT OF COMPUTER  
ENGINEERING

ISTANBUL, FEBRUARY, 2021

NOTICE ON RESEARCH ETHICS AND  
PUBLISHING METHODS

I, KEMAL MAHMUT KAŞGARLI;

- hereby acknowledge, agree and undertake this Master's Degree Thesis that I have prepared is entirely my own work and I have declared the citations from other studies in the bibliography in accordance with the rules;
- this Master's Degree Thesis does not contain any material from any research submitted or accepted to obtain a degree or diploma at another educational institution;
- and that I commit and undertake to follow the "Kadir Has University Academic Codes of Conduct" prepared in accordance with the "Higher Education Council Codes of Conduct".

In addition, I acknowledge that any claim of irregularity that may arise in relation to this work will result in a disciplinary action in accordance with the university legislation.

KEMAL MAHMUT KAŞGARLI

---

05.02.2021

## ACCEPTANCE AND APPROVAL

This study, titled **TWITTER SENTIMENT ANALYSIS VIA MACHINE LEARNING**, prepared by the **KEMAL MAHMUT KAŞGARLI**, was deemed successful with the **UNANIMOUS** as a result of the thesis defense examination held on the **FEBRUARY 5<sup>TH</sup>, 2021** and approved as a **MASTER'S DEGREE THESIS** by our jury.

JURY:

SIGNATURE:

Assoc. Prof. Dr. Taner Arsan (Advisor)	Kadir Has University	_____
Assoc. Prof. Dr. Osman Kaan Erol	Istanbul Technical University	_____
Assoc. Prof. Dr. Habib Şenol	Kadir Has University	_____

I confirm that the signatures above belong to the aforementioned faculty members.

\_\_\_\_\_  
Prof. Dr. Emine Füsun ALİOĞLU  
Director of the School of Graduate Studies  
APPROVAL DATE: / /

## TABLE of CONTENTS

<b>ABSTRACT (English)</b> .....	<b>i</b>
<b>ÖZET</b> .....	<b>ii</b>
<b>ACKNOWLEDGEMENT</b> .....	<b>iii</b>
<b>DEDICATION</b> .....	<b>iv</b>
<b>LIST OF TABLES</b> .....	<b>v</b>
<b>LIST OF FIGURES</b> .....	<b>vi</b>
<b>LIST OF ABBREVIATIONS</b> .....	<b>viii</b>
<b>1. INTRODUCTION</b> .....	<b>1</b>
<b>2. MACHINE LEARNING</b> .....	<b>3</b>
<b>2.1 TYPES OF MACHINE LEARNING</b> .....	<b>3</b>
<b>2.1.1 Supervised Learning</b> .....	<b>4</b>
<b>2.1.2 Unsupervised Learning</b> .....	<b>5</b>
<b>2.1.3 Reinforcement Learning</b> .....	<b>5</b>
<b>2.2 DATA PROCESS STEPS</b> .....	<b>5</b>
<b>3. TEXT MINING &amp; PROCESS</b> .....	<b>7</b>
<b>3.1 TEXT MINING</b> .....	<b>7</b>
<b>3.2 TEXT MINING PROCESS</b> .....	<b>7</b>
<b>3.2.1 Text Pre-processing</b> .....	<b>7</b>
<b>3.2.2 Text Transformation</b> .....	<b>14</b>
<b>3.2.3 Feature Selection</b> .....	<b>14</b>
<b>3.2.4 Data Mining</b> .....	<b>14</b>
<b>3.2.5 Evaluation</b> .....	<b>14</b>
<b>4. TWITTER AND SENTIMENT ANALYSIS</b> .....	<b>15</b>
<b>4.1 TWITTER DATA AND STRUCTURE</b> .....	<b>15</b>
<b>4.1.1 Twitter Structure</b> .....	<b>15</b>
<b>4.1.2 Twitter Data Set</b> .....	<b>16</b>
<b>4.2 FEATURE EXTRACTION</b> .....	<b>18</b>
<b>4.2.1 Bag of Words</b> .....	<b>18</b>

4.2.2 N-Gram .....	19
4.2.3 Tf-Idf Vectors .....	20
4.3 SENTIMENT ANALYSIS .....	24
4.3.1 Sentiment Polarity .....	24
4.3.2 Types of Sentiment Analysis .....	25
4.3.3 Benefit of Sentiment Analysis .....	26
4.3.4 Sentiment Analysis Approaches .....	26
4.3.4.1 Machine Learning Based .....	27
4.3.4.2 Lexicon Based .....	28
4.3.4.2 Hybrid Based .....	30
4.4 LEXICONS .....	30
4.4.1 Vader .....	30
4.4.2 Textblob .....	31
4.4.3 Wordnet .....	33
4.4.4 Sentiwordnet .....	33
5. PROGRAMMING LANGUAGE AND SOFTWARE PACKAGE .....	34
5.1 PYTHON PROGRAMMING .....	34
5.2 PACKAGE TOOLS .....	35
5.2.1 Anaconda .....	35
5.2.2 Jupyter Lab.....	35
5.2.3 Python Libraries .....	35
6. MACHINE LEARNING ALGORITHMS AND ACCURACY .....	37
6.1 SUPPORT VECTOR MACHINE .....	37
6.2 LOGISTIC REGRESSION .....	38
6.3 NAÏVE BAYES .....	39
6.4 RANDOM FOREST.....	41
6.5 XGBOOST.....	41
6.6 ACCURACY .....	42
7. METHODOLOGY .....	43
7.1 PREPROCESSING .....	43
7.2 FREQUENCY OF WORDS IN DATA AND VISULIZATION.....	47
7.3 SENTIMENT ANALYSIS .....	56

<b>7.4 DATA FEATURE EXTRACTION AND MODELLING.....</b>	<b>60</b>
<b>7.5 COMPARISON MACHINE LEARNING ALGORITHM ACCURACY.....</b>	<b>71</b>
<b>8. CONCLUSION .....</b>	<b>80</b>
<b>REFERENCES .....</b>	<b>82</b>
<b>ANNEX A .....</b>	<b>85</b>
<b>A.1 Examples of the Numeric System .....</b>	<b>85</b>
<b>CURRICULUM VITAE .....</b>	<b>86</b>





## TWITTER SENTIMENT ANALYSIS VIA MACHINE LEARNING

### ABSTRACT

People comment on social media platforms, share their feelings and thoughts, and communicate with each other about many issues from the events in the world to the products and services they use. Twitter is one of the most popular social media platforms today. The tweets created by the users of this platform can be a very good data set source for data scientists in the field of Text Mining and in Sentiment Analysis studies in particular. In this thesis, sentiment analysis was performed after the tweet data was passed through the text preprocessing process on the JupyterLab editor on the Anaconda platform with the Python programming language, and the text data was labeled as Negative and Positive by binary classification. Tweet text data was transformed into vectors and processed with feature extraction method such as Bag of Words and Tf-idf, and the accuracy of the classification prediction data was compared with Support Vector Machine, Logistic Regression, Naïve Bayes, Random Forest, Extreme Gradient Boost Machine Learning algorithms.

**Keywords:** Twitter, Text Mining, Sentiment Analysis, Feature Extraction, Machine Learning, Accuracy

# MAKİNE ÖĞRENİMİ YOLUYLA TWITTER DUYGU ANALİZİ

## ÖZET

İnsanlar dünyada yaşanan olaylardan kullandıkları ürün ve hizmetlere kadar bir çok konu hakkında sosyal medya platformlarında yorum yapmakta, duygu ve düşüncelerini paylaşmakta ve birbirleriyle iletişim içinde bulunmaktadır. Twitter günümüzde çok popüler olan sosyal medya platformlarından biridir. Bu platformun kullanıcıları tarafından oluşturulan tweetler Metin Madenciliği alanında ve özelinde Duygu analizi çalışmalarında veri bilimcileri için çok iyi birer veri seti kaynağı olabilmektedir. Bu tez çalışmasında tweet verileri Python programlama dili ile Anaconda platformunda yer alan JupyterLab editörü üzerinde metin ön işleme sürecinden geçirildikten sonra duygu analizleri yapılmış, metin verisi ikili sınıflandırma yapılarak Negatif ve Pozitif olarak etiketlenmiştir. Tweet metin verileri vektörlere dönüştürülerek Bag of Words ve Tf-idf gibi özellik çıkarımı yöntemi ile işlenmiş ve Destek Vektör Makinesi, Lojistik Regresyon, Naïve Bayes, Rastgele Orman, Extreme Gradient Boost Makine Öğrenmesi algoritmaları ile sınıflandırma tahmin verilerinin doğrulukları karşılaştırılmıştır.

**Key words:** Twitter, Metin Madenciliği, Duygu Analizi, Özellik Çıkarımı, Makine Öğrenmesi, Doğruluk

## **ACKNOWLEDGEMENT**

During the execution of this study, my advisor Assoc. Prof. Dr. Taner Arsan who allways support with me by his experience, my mother Mihrinisa Kaşgarlı and my father Prof.Dr. Mahmut Kaşgarlı, my wife Rizvangül Kaşgarlı for showing patience and enduring me during my studies, my daughters Mihray and Kevser for allowing me to work continuously, and our Istanbul University Career Center manager Assoc. Prof. Dr. Aslı Beyhan Acar I would like to thank and everyone who did not spare their little or great assistance during my work.



*To my wife and mother*

## LIST of TABLES

<b>Table 4.1</b> : Feature of Twitter Dataset.....	17
<b>Table 4.2</b> : Example of Rule Based Sentiment Analysis.....	29
<b>Table 4.3</b> : Classification of Sense Class According to Sentiment Score .....	32
<b>Table 6.1</b> : Confusion Matrix .....	42
<b>Table 7.1</b> : Import Data and Set into DataFrame .....	43
<b>Table 7.2</b> : Splitting Data to According to Years of 2017-2018-2019-2020 .....	44
<b>Table 7.3</b> : Dataframe Information .....	45
<b>Table 7.4</b> : Preprocessing Text Data .....	45-46
<b>Table 7.5</b> : Delete Stop Words and Lemmitization Process .....	46
<b>Table 7.6</b> : Number of Data Before and After Preprocessing .....	47
<b>Table 7.7</b> : Python Code of Installation and Usage WordCloud .....	48
<b>Table 7.8</b> : Python Code of Word Frequency.....	50
<b>Table 7.9</b> : Python Code of Sentiment Analysis .....	56
<b>Table 7.10</b> : Sense Class of Tweets according to Years and Whole Data .....	57
<b>Table 7.11</b> : Feature Extraction and Fitting to Train and Test Set .....	60-62
<b>Table 7.12</b> : Python Code of Machine Learning Algorithm Accuracy .....	62-70
<b>Table 7.13</b> : Accuracy Score of Tweets 2017.....	71
<b>Table 7.14</b> : Accuracy Score of Tweets 2018 .....	72
<b>Table 7.15</b> : Accuracy Score of Tweets 2019 .....	73
<b>Table 7.16</b> : Accuracy Score of Tweets 2020 .....	74
<b>Table 7.17</b> : Accuracy Score of Tweets 2017-2020 .....	75
<b>Table 7.18</b> : Accuracy Score of All Tweets .....	76
<b>Table 7.19</b> : Accuracy Score of All Tweets According to %25 Test Size .....	77
<b>Table 7.20</b> : Accuracy Score of All Tweets According to %33 Test Size .....	78
<b>Table 7.21</b> : Accuracy Score of All Tweets According to %40 Test Size .....	79

## LIST of FIGURES

<b>Figure 2.1</b> :Types of Machine Learning .....	4
<b>Figure 2.2</b> :Classification and Regression .....	4
<b>Figure 2.3</b> :Data Process Steps .....	6
<b>Figure 3.1</b> :Conversion of upper case to lower case .....	8
<b>Figure 3.2</b> :Remove Punctuation .....	8
<b>Figure 3.3</b> :Remove Numbers .....	9
<b>Figure 3.4</b> :Tokenization .....	9
<b>Figure 3.5</b> :List of Stop Words .....	10
<b>Figure 3.6</b> :Example of using Stop Words .....	10
<b>Figure 3.7</b> :Example of Stemming .....	11
<b>Figure 3.8</b> :Example of Lemmatization .....	11
<b>Figure 3.9</b> :Tagset for Part of Speech Tagging .....	12
<b>Figure 3.10</b> Example of Part of Speech Tagging .....	12
<b>Figure 3.11</b> Example of Chunking .....	13
<b>Figure 3.12</b> Draw of Chunking Example .....	13
<b>Figure 4.1</b> :Data Types of Twitter Dataset .....	18
<b>Figure 4.2</b> :Bag of Words Example .....	19
<b>Figure 4.3</b> :N-gram Example .....	20
<b>Figure 4.4</b> :Tf-idf Example with Tf-idfVectorizer Function in Python .....	21
<b>Figure 4.5</b> :Use of LabelEncoder and Vectorizer Function .....	22
<b>Figure 4.6</b> :Five Vectorized Sample and Array Form .....	22
<b>Figure 4.7</b> :Tf-idf N-gram Example .....	23
<b>Figure 4.8</b> :Example of Tdf-idf Char Level .....	23
<b>Figure 4.9</b> :Example of Tdf-idf Word Level .....	24
<b>Figure 4.10</b> Example of Sentiment Polarity represent by Sentiment Score .....	25
<b>Figure 4.11</b> Sentiment Analysis Approach .....	27
<b>Figure 4.12</b> Machine Learning Based Sentiment Analysis Flow Chart .....	28
<b>Figure 4.13</b> Machine Learning Based Sentiment Analysis Flow Chart .....	31
<b>Figure 4.14</b> Output of Classification Sense Class .....	33

<b>Figure 6.1</b> : Support Vector Machine Hyperplane .....	38
<b>Figure 6.2</b> : Logistic Regression S-Shape Curve .....	39
<b>Figure 7.1</b> : Output of the Twitter Dataset to Dataframe .....	44
<b>Figure 7.2</b> : Output of the Twitter Data After Preprocessing .....	46
<b>Figure 7.3</b> : Text Data of All Tweets .....	47
<b>Figure 7.4</b> : WordCloud of Tweets 2017 .....	48
<b>Figure 7.5</b> : WordCloud of Tweets 2018 .....	49
<b>Figure 7.6</b> : WordCloud of Tweets 2019 .....	49
<b>Figure 7.7</b> : WordCloud of Tweets 2020 .....	49
<b>Figure 7.8</b> : WordCloud of 2017-2020 Presidency Term .....	50
<b>Figure 7.9</b> : WordCloud of All Tweets .....	50
<b>Figure 7.10</b> : Word Frequence of Tweets .....	51
<b>Figure 7.11</b> : Plot Bar Graph Word Frequency Years 2017-2018-2019-2020.....	52
<b>Figure 7.12</b> : Plot Bar Graph Word Frequency 2017-2020 .....	52
<b>Figure 7.13</b> : Plot Bar Graph of Word Frequency of All Data .....	53
<b>Figure 7.14</b> : Device Used to Post Years 2017-2018-2019-2020 Tweets .....	54
<b>Figure 7.15</b> : Device Used to Post Years 2017-2020 Tweets .....	55
<b>Figure 7.16</b> : Device Used to Post All Tweets .....	55
<b>Figure 7.17</b> : Output of Sentiment Analysis .....	57
<b>Figure 7.18</b> : Sense Class Tweets Years 2017-2018-2019-2020 .....	58
<b>Figure 7.19</b> : Sense Class of 2017-2020 .....	59
<b>Figure 7.20</b> : Sense Class of All Data .....	59

## LIST of ABBREVIATIONS

API	Application Program Interface
FAV	Favorite
FN	False Negative
FP	False Positive
KDD	Knowledge Discovery in Databases
KNN	K-Nearest Neighbor
LR	Logistic Regression
MDP	Markov Decision Process
ML	Machine Learning
NB	Naive Bayes
NER	Named Entity Recognition
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OOP	Object Oriented Programming
PCA	Principal Component Analysis
POS	Part of Speech Tagging
RF	Random Forest
RT	Retweet
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency
TM	Text Mining
TN	True Negative
TP	True Positive
t-SNE	t-Distributed Stochastic Neighbor Embedding
VADER	Valence Aware Dictionary and Sentiment Reasoner
VoC	Voice of Customer
XGBOOST	Extreme Gradient Boosting



## 1. INTRODUCTION

Advances in technology and science, epidemics experienced today have accelerated digitalization. In our digitalizing world, people communicate with each other through digital platforms, communicate and share their feelings and thoughts on all kinds of situations and issues through these platforms. These feelings, thoughts and ideas shared by people on these digital platforms are a very important resource and guide for many people or institutions such as data scientists, academics, companies. One of the most widely used digital social media platforms today is Twitter.

In this thesis, the performance of twitter sentiment analysis and the comparison of the prediction accuracy of emotion classes with machine learning algorithms using feature extraction method. One of the first studies on sentiment analysis was carried out by Pang et al. Using the data received by Imdb in the field of cinema and the positive and negative classification of the comments of the movies. In this study Pang used unigram, bigram, combination of unigram and bigram with POS and machine learning algorithm such as Naïve Bayes, Maximum Entropy and Support Vector Machine. Pang achieved the best accuracy result as 82.9% with Unigram and Support Vector Machine algorithms [1].

Twitter sentiment analysis studies have been carried out by many researchers and academics. Davidov et al. conducted a Twitter sentiment classification study in 2010. In this study, Davidov used hashtags and smiley characters specific to Twitter as well as using the feature vector and observed that it increased the success rate [2].

Although there are many Sentiment analysis studies in the field of English, there is a limited number of studies in Turkish. Some of these studies are Çetin and Amasyalı's work on twitter messages. In this study, Tf, Tf-idf, 1 K-Nearest Neighbor, Decision Tree, Random Forest feature and machine learning algorithms are usedm [3]. Meral and

Diri worked on Twitter data in 2014. In their study, they tagged Tweets, and also used the Char 2-gram and 3-gram features and Support Vector Machine, K- Nearest Neighbor and Naïve Bayes machine algorithms [4].

In this thesis, sentiment analysis of twitter data of Donald Trump, the President of the United States of America, has been made. In the study, textblob lexicon was used, in addition, feature extraction methods such as Count Vector, Tf-idf Word level, Tf-idf N-gram, Tf-idf Char level and machine learning such as Support Vector Machine, Logistic Regression, Naive Bayes, Random Forest and XGBoost algorithms are used. The data set was examined one by one for each year of the presidential term, as presidential term and all data. In addition, the test set was used in different sizes and its effect on accuracy was observed.

## **2. MACHINE LEARNING**

Machine learning (ML) is essentially a sub-branch of computer science after the studies of numerical learning and model recognition in artificial intelligence in 1959. Machine learning and artificial intelligence are often considered together. In some case, they are used interchangeably, but they do not mean the same. An important distinction is that while all machine learning solutions are artificial intelligence but all artificial intelligence solutions are not machine learning.

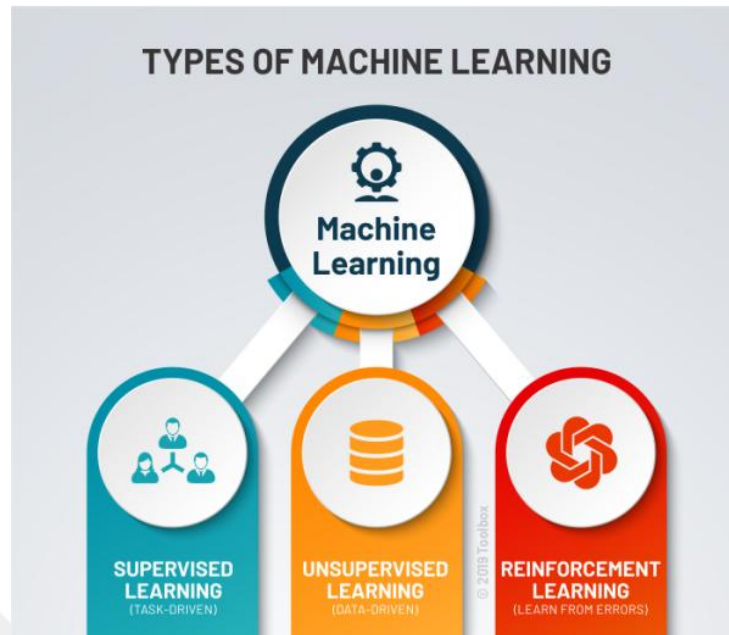
ML is a system that can learn as a structural function and investigate the work and construction of algorithms that can make predictions over data. Such algorithms work by building a model to make data-based predictions and decisions from sample inputs rather than following static program instructions strictly [5]. The definition of the “Machine Learning” by some researcher as follows:

The definition of ML by Kevin Murphy: “as a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty” [6].

The other definition from Thomas W. Edgar, David O. Manz: Machine learning is a field of study that looks at using computational algorithms to turn empirical data into usable models [7].

### **2.1 TYPES OF MACHINE LEARNING**

Machine Learning types vary according to the intended of use and target outputs. As shown as Figure 2.1, ML have three main categories: Supervised Learning, Unsupervised Learning and Reinforcement Learning.

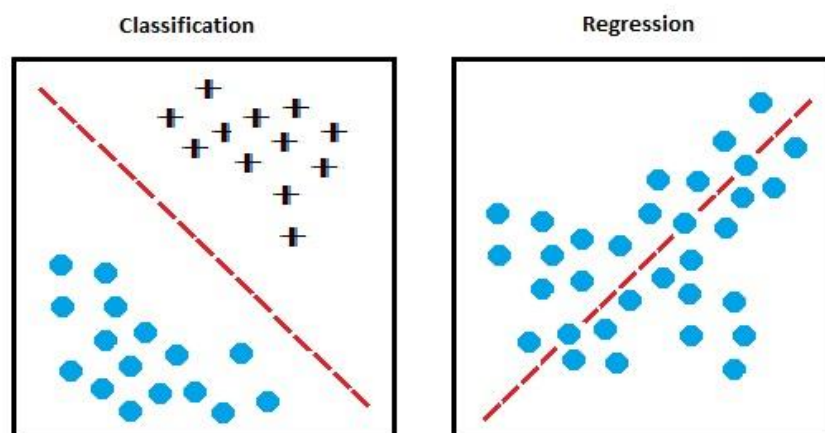


**Figure 2.1. Types of Machine Learning [8]**

### 2.1.1 Supervised Learning

A supervised learning train a model with a dataset which include input(X) and labeled output(Y) variables. According to the model trained in supervised learning, our output data is estimated according to the new input data. Supervised Learning also categorized as Classification and Regression as shown as Figure 2.2.

- Classification: Predict discrete and categorical response value.
- Regression: Predict continuous response value.



**Figure 2.2. Classification and Regression**

Supervised Learning algorithm mostly used are: Linear Regression; Logistical Regression; Random Forest; Gradient Boosted Trees; Support Vector Machines (SVM); Neural Networks; Decision Trees; Naive Bayes; Nearest Neighbor.

### **2.1.2 Unsupervised Learning**

It is a learning method with no data set and no outputs. The purpose of unsupervised learning is to model the structure or distribution that underlies the data to learn more about the data. Unsupervised Learning algorithms are Descriptive Modelling and use unlabeled data.

- Clustering: Grouping according to similarity and relationship among the data given.

Unsupervised Learning algorithm mostly used are: K-means clustering; t-SNE (t-Distributed Stochastic Neighbor Embedding); PCA (Principal Component Analysis); Association rule.

### **2.1.3 Reinforcement Learning**

Reinforcement learning is a machine learning approach inspired by behaviorism that deals with what actions subjects must take in order to achieve the highest amount of reward in an environment. Due to its generality, this problem is also studied in many other branches such as game theory, control theory, operations research, information theory, simulation-based optimization, and statistics. In machine learning, the environment is often modeled as a Markov Decision Process (MDP), in this context many reinforcement learning algorithms use dynamic programming techniques. [8, 9]

## **2.2 DATA PROCESS STEPS**

ML's objective is to derive meaningful info from data. ML has seven steps for data process see in Figure 2.3.



**Figure 2.3. Data Process Steps [10]**

- Data Collection

Gather required data from different source (databases, datasets) for training data. The most known pre-collected data sources are Kaggle, UCI.

- Data Preparation

Clean the noisy and missing data from datasets. Remove the duplicates and correct errors. Handle type conversion and normalization process. Split dataset to training and test data for evaluation.

- Choose a Model

Machine Learning have a lot of model for different purposes. Select the accurcate model which meets business goal.

- Train the Model

The most important part of ML is training model because its aim to improve the increment of the model prediction.

- Evaluate the Model

Evaluating the model with unused data to measure the performance of model.

- Parameter Tuning

Change the default parameters or add parameter to ML model for improved performance.

- Make Predictions

Using new data and test the model performance in real world [10, 11].

## 3. TEXT MINING & PROCESS

### 3.1 TEXT MINING

Text mining (TM) is a data mining which accept data in text as a resource. TM is to obtain structured data through unstructured text data. It is also called as Text Data Mining, Information Extraction or Knowledge Discovery in Databases (KDD). TM aims to classification of texts, clustering, subject extraction from texts (concept / entity recognition), production of granular taxonomy, sentimental analysis, document summarization, entity relationship modeling. TM uses to information retrieval, lexical analysis, word frequency distribution, pattern recognition, tagging, information extraction, data mining and even visualization methods for achieve the above-mentioned goals. TM studies, natural language processing, another field of study in text-based literature (natural language processing, NLP) studies often go hand in hand. Natural language processing studies mostly cover studies based on linguistics under artificial intelligence. TM studies, on the other hand, aim to reach statistical results through text. During text mining studies, feature extraction is often performed using natural language processing.

### 3.2 TEXT MINING PROCESS

#### 3.2.1 Text Pre-processing

Text Pre-processing is a process which convert to data a standart format. Firstly the unstructured text must be clean from noisy and missing data. Text preprocessing of text mining have few steps such as

- *Text Normalization* (remove number, punctuations, white spaces, stop words and convert lower case)
- *Tokenization*
- *Stemming*
- *Lemmatization*
- *Part-of-speech Tagging* (POS)

- *Chunking*
- *Named Entity Recognition (NER).*

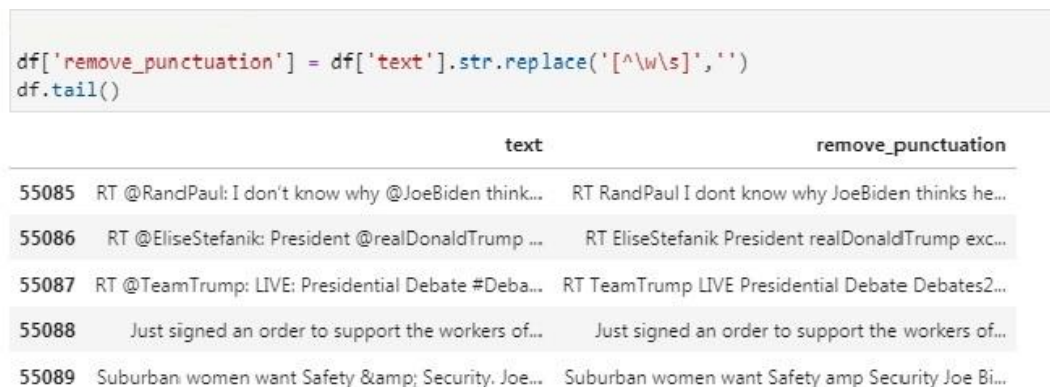
### Text Preprocessing Steps

➤ **Lower Case:** Converting upper case to lower case is the common text preprocessing technique. It is necessary to convert the input text into the same format so that it is considered the same as "TEXT" and "text" as shown in Figure 3.1



**Figure 3.1. Conversion of upper case to lower case**

➤ **Remove Punctuation:** Removing punctuation is a kind of text preprocessing technique shown in Figure 3.2. This process standartize the text so that ‘warning!’ converted to ‘warning’. String.punctuation contains the following symbols: !"#\$%&'\()\*+,-./:;<=>?@[\\]^\_`{|}~` Developer can add or remove punctuations according to case.



**Figure 3.2. Remove Punctuation**



➤ **Remove Numbers:** It is a preprocessing technique which remove the numeric data from text if its not related to your analysis shown in Figure 3.3.

```
df['text_removed_number'] = df['text'].str.replace('\d', '')
df.iloc[8:11]
```

	text	text_removed_number
8	httpstco4qwckqoiow	httpstcoqwckqoiow
9	httpstcovleu8yyovv	httpstcovleuyyovv
10	httpstcoz5crqho8vg	httpstcozcrqhovg

**Figure 3.3. Remove Numbers**

➤ **Tokenization:** is a preprocessing technique which split the text smaller pieces called ‘tokens’ shown in Figure 3.4. The tokens include word, number, punctuation mark and others.

```
import pandas as pd
import nltk

df = pd.DataFrame({'sentences': ['Example of Tokenization. important part of 3 section!', 'Please call for details 000555448899.', 'Good idea, i like it...']})
df['tokenized_sentences'] = df.apply(lambda row: nltk.word_tokenize(row['sentences']), axis=1)

df.head()
```

	sentences	tokenized_sentences
0	Example of Tokenization. important part of 3 s...	[Example, of, Tokenization, ,, important, part...
1	Please call for details 000555448899.	[Please, call, for, details, 000555448899, .]
2	Good idea, i like it...	[Good, idea, ,, i, like, it, ...]

**Figure 3.4. Tokenization**

➤ **Remove Stopwords:** “Stop words” are the foremost common words in a language like “a”, “an”, “the”. Stop words don’t have important meaning or valuable information in text so they can remove from text. Natural Language Toolkit NLTK also use for removing the stop words from text.

There are lists of Stop words compiled for different languages. Developer use NLTK packages safely which include list of stop words shown in Figure 3.5.

```
print(stopwords.words('english'))

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours',
'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'its
elf', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these',
'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doin
g', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about',
'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'o
ut', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'al
l', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so',
'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o',
're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'has
n', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'sha
n', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

**Figure 3.5. List of Stop Words**

The sentences cleaned from stop words shown in Figure 3.6 .

```
tweets = [('I love this game', 'positive'),
          ('Its my favorite team', 'positive'),
          ('I am very glad to meet you', 'positive'),
          ('I hate playing basketball', 'negative'),
          ('I dont like watching TV', 'negative')]

test = pd.DataFrame(tweets)
test.columns = ["tweet", "label"]
test["tweet"] = test["tweet"].str.lower().str.split()

from nltk.corpus import stopwords
stop = stopwords.words('english')

test['tweet'].apply(lambda x: [item for item in x if item not in stop])

0          [love, game]
1    [favorite, team]
2          [glad, meet]
3    [hate, playing, basketball]
4    [dont, like, watching, tv]
Name: tweet, dtype: object
```

**Figure 3.6. Example of using Stop Words**

- **Stemming:** is a preprocessing technique which convert the words to its stem or root form shown in Figure 3.7. Stemming has two algorithm:
  - Porter Stemming Algorithm: Removes common morphological and inflexional endings from words.
  - Lancaster Stemming Algorithm: Aggressive stemming algorithm.

```

from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
stemmer= PorterStemmer()
input_str= "Republicans and Democrats have both created our economic problems."
input_str=word_tokenize(input_str)
for word in input_str:
    print(stemmer.stem(word))

```

```

republican
and
democrat
have
both
creat
our
econom
problem

```

**Figure 3.7. Example of Stemming**

- **Lemmatization:** is a preprocessing technique which reduce inflectional forms to a common base form. Lemmatization doesn't simply chop off inflections and it uses lexical knowledge bases to get the correct base forms of words shown in Figure 3.8. Lemmatization tools are presented in NLTK (WordNet Lemmatizer) and TextBlob and other libraries.

```

from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
lemmatizer=WordNetLemmatizer()
input_str="Republicans and Democrats have both created our economic problems."
input_str=word_tokenize(input_str)
for word in input_str:
    print(lemmatizer.lemmatize(word))

```

```

Republicans
and
Democrats
have
both
created
our
economic
problem

```

**Figure 3.8. Example of Lemmatization**

- **Part of Speech Tagging (POS):** is a preprocessing technique which assign tag shown in Figure 3.9, to each word of text according to its definition and context. The tags are change according to words structure( Verbs, Noun, Adjective, Adverbs etc.) on text and labeled as abbreviations as NN, JJ, VBS, DT, IN etc.

The tools includes POS taggers are: NLTK, spaCy, TextBlob, Pattern, Stanford CoreNLP, Apache OpenNLP.

Main Tags	Representation
Noun	NN
Noun Location	NST
Proper Noun	NNP
Pronoun	PRP
Compound Words	XC
Demonstration	DEM
Post Position	PSP
Conjuncts	CC
Verb	VM
Adverb	RB
Particles	RP
Adjectives	JJ
Auxiliary Verb	VAUX
Negation	NEG
Quantifiers	QF
Cardinal	QC
Ordinal	QO
Question Words	WQ
Intensifiers	INTF
Interjection	INJ
Reduplication	RDP
Unknown Words	UNK
Symbol	SYM

**Figure 3.9. Tagset for Part of Speech Tagging**

Each word of text assigned tag according to its definition and context shown in Figure 3.10.

```
import nltk
nltk.download('averaged_perceptron_tagger')
input_str= "Republicans and Democrats have both created our economic problems.Problems can solve in short time."
from textblob import TextBlob
result = TextBlob(input_str)
print(result.tags)

[('Republicans', 'NNPS'), ('and', 'CC'), ('Democrats', 'NNPS'), ('have', 'VBP'), ('both', 'DT'), ('created', 'VBN'), ('our', 'PRP$'), ('economic', 'JJ'), ('problems.Problems', 'NNS'), ('can', 'MD'), ('solve', 'VB'), ('in', 'IN'), ('short', 'JJ'), ('time', 'NN')]
```

**Figure 3.10. Example of Part of Speech Tagging**

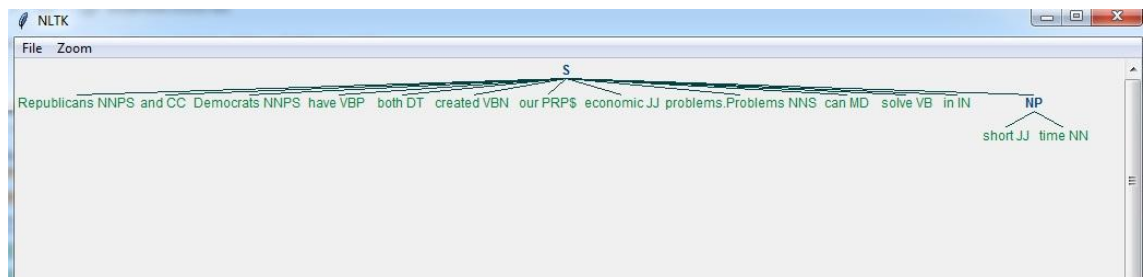
➤ **Chunking:** is a natural language process which identifies the words of text as Noun, Verbs, Adjective and links them higher order unit according to their grammatical meanings shown in Figure 3.11 and draw of Chunking shown in Figure 3.12. The tools include Chunking are: NLTK, TreeTagger chunker, Apache OpenNLP, General Architecture for Text Engineering (GATE).

```
reg_exp = "NP: {<DT>?<JJ>*<NN>}"
rp = nltk.RegexpParser(reg_exp)
result = rp.parse(result.tags)
print(result)
```

```
(S
  Republicans/NNPS
  and/CC
  Democrats/NNPS
  have/VBP
  both/DT
  created/VBN
  our/PRP$
  economic/JJ
  problems.Problems/NNS
  can/MD
  solve/VB
  in/IN
  (NP short/JJ time/NN))
```

```
result.draw()
```

**Figure 3.11. Example of Chunking**



**Figure 3.12. Draw of Chunking Example**

➤ **Named Entity Recognition:** is points to discover named substances in content and classify them into pre-defined categories as names of people, areas, organizations, times, etc. Named-entity acknowledgment instruments: cNLTK, spaCy, Common Engineering for Content Designing (Door) — ANNIE, Apache OpenNLP, Stanford CoreNLP, DKPro Center, MITIE, Watson Common Dialect Understanding.

### 3.2.2 Text Transformation

This approach specifically requires the representation of the document by the text it contains and the number of occurrences. For this phase, there are mainly two strategies.

- *Bag of Words:* A text is represented as a bag of words (multi-set), ignoring grammar and even word order, but maintaining multiplicity.
- *Vector Space:* A document is converted into a vector of word-derived index terms in this model. A term that appears in their text corresponds to each dimension of the vector.

### 3.2.3 Feature Selection

Selection of features is also known as selection of attributes or variable selection. Most of the information for your prediction variables is provided by the selection of the most relevant characteristics from the available variables. The irrelevant characteristics will increase the complexity and decrease the accuracy of analysis.

### 3.2.4 Data Mining

We combine the text mining method with conventional techniques for data mining. Once the data is organized after the above steps, the classic techniques of data mining are applied to the data to extract the data. Such approaches include classification, clustering, regression, outer, sequential patters, rules of prediction and association.

### 3.2.5 Evaluation

We get an end result after the data mining techniques are applied. This outcome should be measured and tested for the accuracy of the prediction [12].

## 4. TWITTER AND SENTIMENT ANALYSIS

### 4.1 TWITTER DATA AND STRUCTURE

In the digitalized world, people can communicate with each other via social media platforms such as Twitter, Facebook and Instagram. The social media platforms allow people to share their feelings and thoughts, share images, files, content, make friends and interact each other in virtual environment.

Twitter is also popular social media platform which people express their feelings and thoughts about actual and trend topics. The platform allows to send a bulk text of 280 characters and to share videos, images and links. Twitter has 1.3 billion user accounts and 330 million active users, users send 550 million tweets and it's nearly 350.000 per minute. The platform is used by many users from famous people to politicians, athletes, artists to big companies with different aims.

Companies use the Twitter as a digital marketing tool and an average of 164 million advertisements are shown every day on Twitter. They also analyze the twitter data for user comments and feedbacks about their product or services. Companies get an idea with these comments, likes, and feedbacks on Twitter and take action accordingly.

Politicians also have the chance to convey their ideas, thoughts and actions to a large mass via Twitter. August 2020 the most followed on Twitter is the 44th president of the United States Barack Obama with 121 million followers.

In this thesis, sentiment analysis of the twitter data of Donald John Trump, the 45th President of the United States of America, who has 85 million Twitter followers, has been made. Twitter is an important data source that contains a wide range of data for the analysis of the opinions and thoughts of the societies against the events happening in the world [13].

#### 4.1.1 Twitter Structure

**Tweet:** One of Twitter's famous features is to write posts with 280 characters. The name of this post is Tweet.

**Retweet (RT):** RT used at the beginning of tweets is used to indicate that the tweet was taken from another user and published in the user's own timeline.

**Hashtag (#):** It means label and indicated with a # sign. It is used to indicate which topic the tweet is related and to summarize the final emotion or thought within the tweet. In this way tweet reach target group. If user includes #scholarship hashtag to their tweets, the Twitter user reaches that tweets who follow the #scholarship hashtag.

**Mention (@):** Mentioning Twitter users by their username (@Username) and to notify them through the notification mechanism provided by Twitter.

**Favorite (FAV):** This is the process of liking the posted tweet. It is accomplished by clicking the heart sign under the tweet.

**Trend Topic (TT):** Refers to the popular topic. If a # hashtag used for many times and twitter users tweet about that hashtag, it can be TT. The top 10 titles are qualified as TT.

#### **4.1.2 Twitter Data Set**

Twitter Application Program Interface (API) is used to extract data from Twitter. The Tweepy is Python library which use for extract real-time data from Twitter. Tweepy used to connect with Twitter API.

***Installing Tweepy by pip command:*** !pip install tweety

***Installing Tweepy on Anaconda command:*** conda install –c conda-forge tweepy

Create an APP for getting tweets from Twitter, user need a Twitter Developer account and API keys.

The steps are:

Create a Twitter account -> Apply Twitter Developer Account -> Create a APP ->

Generate a API key

Key include



- Consumer Key
- Consumer Secret Key
- Access Token
- Access Token Secret

Recently, it has become difficult to get a Twitter Development account and even if the account is created, due to some restrictions, not much data can be accessed. In this thesis Donald John Trump tweets data was collected from <https://www.thetrumparchive.com/> website at 06-12-2020. Every 60 seconds, this website checks Twitter and records every Trump tweet into a database. All available tweets were collected and added to the database for perpetuity until the site was launched in 2016.

The tweet data include 55090 entries and eight columns. Columns name are “id, text, isRetweet, isDeleted, device, favorites, retweets, date”. In this thesis the “text” column data used for feature extraction and sentiment analysis. Features of Twitter Dataset shown in Table 4.1.

**Table 4.1. Feature of Twitter Dataset**

<b>Name of Columns</b>	<b>Description</b>
<b>Id</b>	Tweet Identification Number
<b>Text</b>	Tweet’s text
<b>isRetweet</b>	Retweeted or Not Retweeted (True=t/False=f)
<b>isDeleted</b>	Deleted or Not Deleted (True=t/False=f)
<b>Device</b>	The name of device written in this section which post the tweets such as Iphone, Ipad, Tweetdeck.
<b>Favorites</b>	Number of tweet’s like which done by clicking heart sign.
<b>Retweets</b>	Number of tweet repost
<b>Date</b>	Date of posted tweet

As shown in Figure 4.1, Id, favorites and retweets are numerical data which data type is int64 and text, isRetweet, isDeleted, device and date are object.

```
df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55090 entries, 0 to 55089
Data columns (total 8 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   id           55090 non-null  int64
1   text         55090 non-null  object
2   isRetweet    55090 non-null  object
3   isDeleted    55090 non-null  object
4   device       55090 non-null  object
5   favorites    55090 non-null  int64
6   retweets     55090 non-null  int64
7   date         55090 non-null  object
dtypes: int64(3), object(5)
memory usage: 3.4+ MB
```

**Figure 4.1. Data Types of Twitter Dataset**

## 4.2 FEATURE EXTRACTION

### 4.2.1 Bag Of Words

Converts text file to vector according to frequency(count) of each word. Its useful for feature extraction and text mining analysis. In Python, scikit-learn library has a function CountVectorizer() which uses for that process. Count Vectors create a matrix which columns are represent as a unique word in a text and the row represent as each text sample.

Document1: Machine Learning has Supervised Learning

Document2: Machine Learning has also Unsupervised Learning

Bag of Words separate unique word in documents and create a list of unique words. Documents have 6 unique words. It means documents have 6 dimension or feature. Finally a creation vector of each document according to frequency of unique words shown in Figure 4.2.

```

from sklearn.feature_extraction.text import CountVectorizer

text = ['Machine Learning has Supervised Learning', 'Machine Learning has also Unsupervised Learning']
count_vector = CountVectorizer()
A = count_vector.fit_transform(text)

import pandas as pd

df = pd.DataFrame()
df['document_No'] = count_vector.get_feature_names()
df['document1'] = A.toarray()[0]
df['document2'] = A.toarray()[1]
df.set_index('document_No', inplace=True)
print(df.T)

```

document_No	also	has	learning	machine	supervised	unsupervised
document1	0	1	2	1	1	0
document2	1	1	2	1	0	1

**Figure 4.2. Bag of Words Example**

### 4.2.2 N-Gram

N-gram express the N words sequence in a text. If N is 1 it express as Unigram, N is 2 it express as Bigram and N is 3 it express as Trigram.

Text: “Machine Learning has also Unsupervised Learning”

**Unigram:** ‘Machine’, ‘Learning’, ‘has’, ‘also’, ‘Unsupervised’, ‘Learning’

**Bigram:** ‘Machine Learning’, ‘Learning has’, ‘has also’, ‘also Unsupervised’, ‘Supervised Learning’

**Trigram:** ‘Machine Learning has’, ‘Learning has also’, ‘has also Unsupervised’, ‘also Unsupervised Learning’ [14] shown in Figure 4.3.

```

from nltk import ngrams

text = 'Machine Learning has also Unsupervised Learning'

Ngram1 = ngrams(text.split(' '), n=2)
Ngram2 = ngrams(text.split(' '), n=3)

for x in Ngram1:
    print(x)

for y in Ngram2:
    print(y)

('Machine', 'Learning')
('Learning', 'has')
('has', 'also')
('also', 'Unsupervised')
('Unsupervised', 'Learning')
('Machine', 'Learning', 'has')
('Learning', 'has', 'also')
('has', 'also', 'Unsupervised')
('also', 'Unsupervised', 'Learning')

```

**Figure 4.3. N-gram Example**

### 4.2.3 Tf-Idf Vectors (Term Frequency - Inverse Document Frequency)

Tf-idf is way to represent a document as feature vector. It measure the importance a particular words in a document or a corpus as shown in Figure 4.4. Three types of Tf-idf vector as below:

- TF-IDF Word Level
- TF-IDF N-gram Level
- TF-IDF Char Level

**TF(t):** (Frequency of a t term in a document)/( Total number of terms in the document)

**IDF(t):**  $\log_e(\text{Total number of documents} / \text{Number of documents containing t-term})$  [15]

The calculation of inverse document frequency and term frequency-inverse document frequency is given in equation (4.1).

$$idf(d, t) = \log \frac{n}{1 + df(t)} \quad (4.1)$$

$$tf - idf(d, t) = tf(d, t) \cdot idf(t)$$

```

from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd

text = ['Machine Learning has Supervised Learning',
        'Machine Learning has also Unsupervised Learning',
        'Machine Learning use the Data Mining technique']

tfidf = TfidfVectorizer()
vector = tfidf.fit_transform(text)

df = pd.DataFrame()
df['document_no'] = tfidf.get_feature_names()
df['document1'] = vector.toarray()[0]
df['document2'] = vector.toarray()[1]
df['document3'] = vector.toarray()[2]
df.set_index('document_no', inplace=True)
print(df.T)

```

document_no	also	data	has	learning	machine	mining
document1	0.000000	0.000000	0.417233	0.648038	0.324019	0.000000
document2	0.480984	0.000000	0.365801	0.568154	0.284077	0.000000
document3	0.000000	0.41894	0.000000	0.247433	0.247433	0.41894

document_no	supervised	technique	the	unsupervised	use
document1	0.548612	0.000000	0.000000	0.000000	0.000000
document2	0.000000	0.000000	0.000000	0.480984	0.000000
document3	0.000000	0.41894	0.41894	0.000000	0.41894

**Figure 4.4. Tf-idf Example with Tf-idfVectorizer Function in Python**

In thesis after preprocessing and sentiment analysis of twitter data, Count Vector and Tf-idf Word level, Tf-idf N-gram level, Tf-idf Char level feature extraction were used for comparing sentiment analysis with machine learning algorithm accuracy score. The entire twitter data split into two parts as training set and test set which were called also as dependent variable and independent variable respectively. The process implemented by “model\_selection.train\_test\_split” function which is a member of sklearn library.

The independent variable “sense\_class” is classified as “Positive, Negative, Neutral”. The preprocessing.LabelEncoder () function was used for converting the value of “sense\_class” string value to numeric value. Encoder was applied for train set and test set at(independent variable) same time. CountVectorizer applied to train and test data (dependent variable) shown in Figure 4.5. The unique word between 5th and 10th element of listed as a feature below. Array () function shows the sparse matrix shown in Figure 4.6.

```
vectorizer.get_feature_names()[5:10]
['abedin', 'abolish', 'abolishing', 'abrams', 'absolute']

x_train_count.toarray()
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       ...,
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

**Figure 4.5. Use of LabelEncoder and Vectorizer Function**

```
from textblob import TextBlob
from sklearn import model_selection, preprocessing, linear_model, naive_bayes, metrics
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn import decomposition, ensemble

import pandas, xgboost, numpy, textblob, string
from keras.preprocessing import text, sequence
from keras import layers, models, optimizers

from warnings import filterwarnings
filterwarnings('ignore')

train_x, test_x, train_y, test_y = model_selection.train_test_split(df18["text"],
                                                                    df18["sense_class"],
                                                                    random_state = 1)

encoder = preprocessing.LabelEncoder()
train_y = encoder.fit_transform(train_y)
test_y = encoder.fit_transform(test_y)

from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
vectorizer.fit(train_x)

x_train_count = vectorizer.transform(train_x)
x_test_count = vectorizer.transform(test_x)
```

**Figure 4.6. Five Vectorized Sample and Array Form**

N-gram level TfidfVectorizer applied to train and test data (dependent variable) n-gram\_range is equal to (2,3) shown in Figure 4.7.

```
from sklearn.feature_extraction.text import TfidfVectorizer

tf_idf_ngram_vectorizer = TfidfVectorizer(ngram_range = (2,3))
tf_idf_ngram_vectorizer.fit(train_x)

TfidfVectorizer(ngram_range=(2, 3))

x_train_tf_idf_ngram = tf_idf_ngram_vectorizer.transform(train_x)
x_test_tf_idf_ngram = tf_idf_ngram_vectorizer.transform(test_x)

tf_idf_ngram_vectorizer.get_feature_names()[24:40]
```

```
['abe heading back',
 'abe hideki',
 'abe hideki matsuyama',
 'abe japan',
 'abe japan easy',
 'abe japan president',
 'abe japan representative',
 'abe japanese',
 'abe japanese people',
 'abe noh',
 'abe noh korea',
 'abe united',
 'abe united state',
 'abe yield',
 'abe yield many',
 'ability mayor']
```

**Figure 4.7. Tf-idf N-gram Example**

Char level level TfidfVectorizer applied to train and test data (dependent variable) shown in Figure 4.8.

```
tf_idf_chars_vectorizer = TfidfVectorizer(analyzer = "char", ngram_range = (2,10))
tf_idf_chars_vectorizer.fit(train_x)

TfidfVectorizer(analyzer='char', ngram_range=(2, 10))

x_train_tf_idf_chars = tf_idf_chars_vectorizer.transform(train_x)
x_test_tf_idf_chars = tf_idf_chars_vectorizer.transform(test_x)

tf_idf_chars_vectorizer.get_feature_names()[10:17]
```

```
[' aba', ' aban', ' aband', ' abando', ' abandon', ' abandone', ' abandoned']
```

**Figure 4.8. Example of Tdf-idf Char Level**



Word level TfidfVectorizer applied to train and test data (dependent variable) shown in Figure 4.9 [14, 15].

```
tf_idf_word_vectorizer = TfidfVectorizer()  
tf_idf_word_vectorizer.fit(train_x)  
  
TfidfVectorizer()  
  
x_train_tf_idf_word = tf_idf_word_vectorizer.transform(train_x)  
x_test_tf_idf_word = tf_idf_word_vectorizer.transform(test_x)  
  
tf_idf_word_vectorizer.get_feature_names()[10:17]  
  
['abound', 'abroad', 'absolute', 'absolutely', 'abt', 'abuse', 'abused']
```

**Figure 4.9. Example of Tdf-idf Word Level**

### **4.3 SENTIMENT ANALYSIS**

Sentiment analysis known also Opinion Mining is a Natural Language Processing technique which detects whether text data's sentiments are positive, negative or neutral. Sentiment analysis is basically a text processing operation and it aims to determine the sentiment class of given text what wants to express. The first studies of sentimental analysis were based on sentimental polarity and aims to classify the given text as positive, negative and neutral [16].

#### **4.3.1 Sentiment Polarity**

Text-based sentiment analysis is based on calculating the polarity of the words or word groups that make up the text. Available dictionaries define the polarity of words and word groups as a decimal value in the range [-1, +1]. The proximity to -1 indicates the negativity of the word, and the proximity to +1 indicates the positivity of the word.

Sentiment analysis is used in many areas according to its need and purpose. The most important areas where sentiment analysis are used Social media monitoring, Customer support, Customer feedback, Brand monitoring and reputation management, Voice of customer (VoC) Voice of employee, Product analysis Market research and Competitive research. In this thesis the Sentiment polarity are calculated and named as "sentiment\_score" shown in Figure 4.10.



	text	sentiment_score
3465	said personally directed fix unmasking process...	-0.215000
3466	year low illegal immigration year	-0.250000
3467	house vote controversial fisa act today act ma...	-0.105556
3468	new quinnipiac poll people feel economy excell...	0.612121
3469	disproven paid democrat dossier used spy trump...	-0.085417
3470	good news toyota mazda announce giant new hunt...	0.167273
3471	cutting tax simplifying regulation make americ...	0.100000
3472	today great honor welcome prime minister erna ...	0.800000
3473	want thank working tirelessly behalf country y...	0.800000

**Figure 4.10. Example of Sentiment Polarity represent by Sentiment Score**

### 4.3.2 Types of Sentiment Analysis

**Aspect-based Sentiment Analysis:** is aimed to analyze the sentiment according to special aspect, feature of a product or subject. It provides more detailed and accurate information to be obtained from the text data by dividing into small pieces considering the subject / emotion integrity.

The product is very high quality and excellent but it was delivered to me too late.

There are more than one emotion and subject in this sentence. For this reason, if the whole sentence is treated as a single statement, it can be chosen as the dominant class, negative or positive, and information may be lost. Calling this content neutral, which contains expressions that clearly contain subjective judgments, can also lead to major mistakes. The following information can be obtained from this sentence with Aspect-based sentiment analysis.

**1. Sentence:** The product is very high quality and excellent

**Subject:** Product **Sentiment:** Positive

**2. Sentence:** It was delivered to me too late

**Subject:** Delivery Time **Sentiment:** Negative

**Fine-grained Sentiment Analysis:** In some cases it may be insufficient to classify text data only as negative or positive. Fine-grained sentiment analysis is used if the polarity sensitivity is important for your business when scaling such as customer satisfaction. Polarity divided 5 categories as below and ranked by a fivestar rate.

Very Positive: 5

Positive: 4

Neutral: 3

Negative: 2

Very Negative: 1

**Emotional Sentiment Analysis:** explores the emotion felt in more detail, not the standard positive, negative, and neutral classes. Uses emotion classes such as happiness, frustration, anger, and sadness. In systems using emotion determination feature, although the attitude of the user is examined positively or negatively thanks to these emotion classes, the mood can be analyzed more clearly and appropriate actions can be taken.

#### **4.3.3 Benefit of Sentiment Analysis**

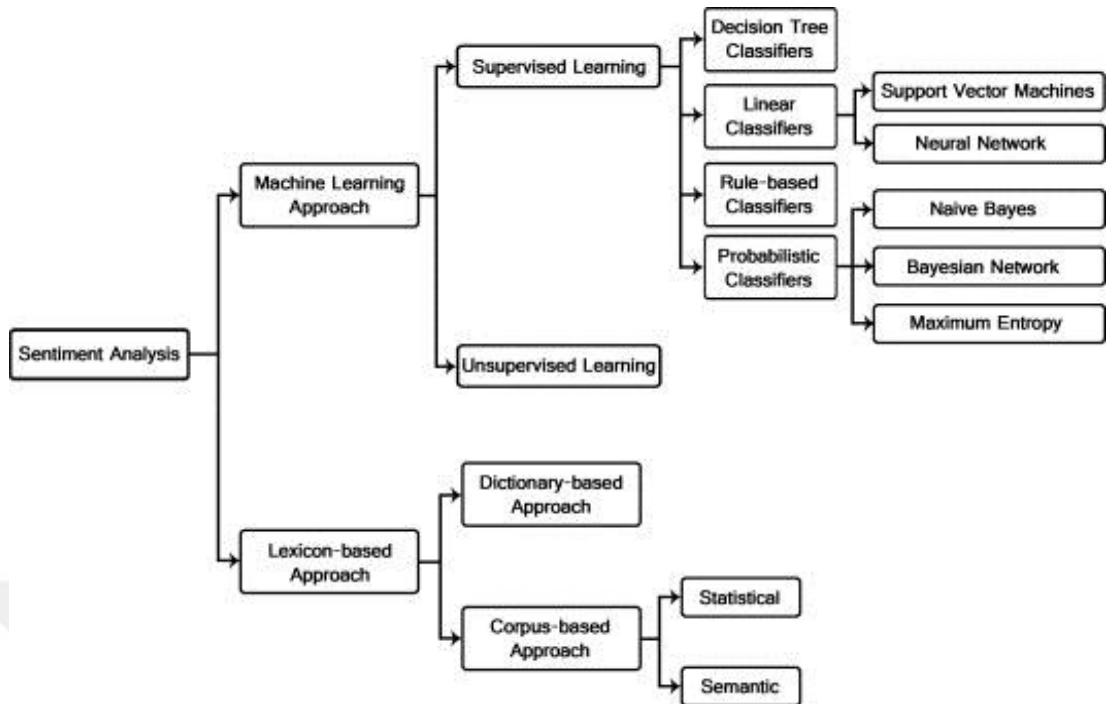
*Data Scalability:* It is impossible process huge amount of data manually such as tweets, customer comments and feedback etc. Sentiment analysis allows data to be scaled and processed in an efficient and cost-effective manner.

*Real-Time Analysis:* Sentiment analysis can be used in real time to identify critical information during a given situation. It is of great importance to determine the situation and intervene quickly.

*Consistent Criteria:* It's difficult to make a consistent assessment when determining the sentiment of a text manually because of subjectivity. Applying centralized sentiment analysis with same criteria to the data increase consistency and reduce errors [17].

#### **4.3.4 Sentiment Analysis Approaches**

Sentiment Classification techniques are shown in Figure 4.11.



**Figure 4.11. Sentiment Analysis Approach [18]**

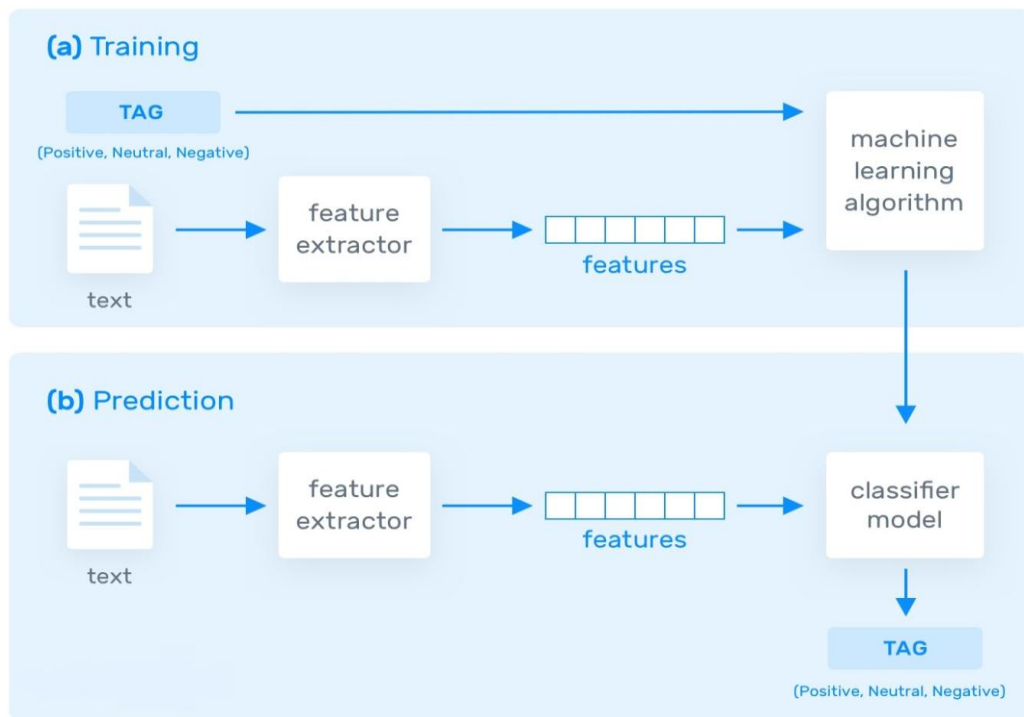
#### 4.3.4.1 Machine Learning Based

Sentiment classification made by Machine Learning algorithm. It has two parts Training and Prediction process shown in Figure 4.12.

*Training:* Model learns to combine the input data(text) with corresponding output(tags) according to the test sample training. Inputs converted to feature vectors by feature extractor. The feature vector and tags are fed into machine learning algorithm to generate model.

*Prediction:* Input data(text) converted to feature vector by feature extractor. Then the feature vector fed into model which generates the tags.

Classification models are used Machine Learning Based Sentiment Analysis are Support Vector Machine, Naive Bayes, Linear Regression, Neural Network [19].



**Figure 4.12. Machine Learning Based Sentiment Analysis Flow Chart [19]**

#### 4.3.4.2 Lexicon Based

Lexicon Based approach used Natural Language Processing(NLP) technique and tools to find patterns in text data and inferring the emotion of the given piece of information which could be classified categories as positive, negative or neutral. Lexicon Based approach used Lexicons(dictionary) which include list of words, expressions and sense of terms. Lexicon Based approach dont need to a labeled or tagged data like Machine Learning Based approach. Disadvantages of Lexicon Based approach: although it need NLP tool and lexicons it does not intend for a specific field of study. Fails to detect indirectly expressed emotion polarities.

Lexicon Based approach divided also 3 categories such as Conditional Random Fields, Dependency Tree and Rule Based Approach. In this thesis, the Ruled Based approach and lexicons structures will be examined.

*Ruled Based Approach*

Rule Based approach uses a set of human-crafted rules to identify the subject of an opinion, subjectivity and polarity. Ruled Based approach also use the NLP technique, tools and lexicons shown in Table 4.2.

**Table 4.2. Example of Rule Based Sentiment Analysis**

Example:

“Dana has a good idea .” →

TOKENIZE	PREPROCESSING
Dana	
Has	Stopwords
A	Stopwords
Good	
Idea	
.	Punctuation

TOKENIZE
Dana
has
a
good
idea
.

Dana	Neutral
good	Positive
Idea	Neutral

The given text data is often unstructured and this data needs to be cleaned and extracted from its unnecessary elements. For this purpose, NLP techniques are applied and text preprocessing processes performed according to the rule.

The process step are shown as below:

- Select the NLP techniques and lexicons.

Technique: Tokenization, Stemming, Lemmatization, Part of Speech tagging.

Lexicon: Vader(Valence Aware Dictionary and Sentiment Reasoner), TextBlob, Sentiwordnet, Wordnet.

- Preprocessing the given text.

Split to text data to individual words which called as tokenize. Remove stopwords such as “a,an, the” because of the those words not carry significant meaning. Remove the numeric value and punctuation from text data. Remove unnecessary symbol and character from data.

After preprocessing the text data the lexicon applied to the data for measuring sentiment analysis. The sentiment of the data shown as Positive because the “good” is positive word in used lexicon.

#### **4.3.4.3 Hybrid Based**

Hybrid Based is combination of Rule Based Approach and Machine Learning Based Approach. The best benefit of Hybrid Based Approach to reach more accurate results.

### **4.4 LEXICONS**

There are many lexicons such as Vader, TextBlob, WordNet, SentiWordNet, SenticNet, AFINN, Sentistrength, NoSlang.com

#### **4.4.1 Vader (Valence Aware Dictionary and Sentiment Reasoner)**

Generally used analyzing for social media sentiment. Vader is sensitive both polarity(Negative, Positive) and intensity(strength) of emotion. Its available in NLTK packages and can apply for unlabeled text directly. The sum of the intensity scores of

each word in a sentence gives the sentiment score of that sentence shown in Figure 4.13. Vader is also smart enough to realize that a phrase like "don't like" is a negative sentence. It understands the emphasis of punctuation and capitalization. The intensity is ranged -4 to +4 value.

- 4:** Extremely Negative
- +4:** Extremely Positive
- 0:** Neutral or N/A

Word	Sentiment	Valence/Intensity
okay	Positive	0.9
good	Positive	1.9
great	Positive	3.1
horrible	Negative	-2.5
:(	Negative	-2.2
Sucks (or the slang: sux)	Negative	-1.5

**Figure 4.13. Machine Learning Based Sentiment Analysis Flow Chart [20]**

#### 4.4.2 TextBlob

TextBlob is a Natural Language Processing (NLP) python library. To accomplish its functions, TextBlob actively uses the Natural Language ToolKit (NLTK). NLTK is a library that allows users to quickly access a wide range of lexical tools and to deal with categorization, classification, and many other activities. TextBlob is a simple library supporting complicated textual data analysis and operations [21].

TextBlob's sentiment property returns polarity and subjectivity given text data. The polarity range is between -1.0 and 1.0 If the value is closer -1.0 its a negative but if it is closer +1.0 its polarity positive. The subjectivity of a instance range between 0.0-1.0. If the instance's subjectivity value closer to 0.0 its very objective but if it is closer +1.0 its very subjective [19-21].

Installing the TextBlob code in Jupyter Notebook:  

```
!pip install textblob
```

After installation the library must import/add by:

```
from textblob import TextBlob
```

In this thesis TextBlob is used and the text of tweets labeled as “Negative, Positive and Neutral” in “sense\_class” column and polarity of the text data written in the column “sentiment\_score”. The code of the project shown in Table 4.3 and output of Classification Sense Class shown in Figure 4.14.

**Table 4.3. Classification of Sense Class According to Sentiment Score**

```
from textblob import TextBlob

def sentiment_scoring(df18):

    text = df18["text"]

    for i in range(0,len(text)):

        textB = TextBlob(text[i])

        sentiment_score = textB.sentiment.polarity

        df18.at[i, 'sentiment_score']=sentiment_score

        if sentiment_score <0.00:

            sense_class = 'Negative'

            df18.at[i, 'sense_class' ]=sense_class

        elif sentiment_score >0.00:

            sense_class = 'Positive'

            df18.at[i, 'sense_class' ]=sense_class

        else:

            sense_class = 'Neutral'

            df18.at[i, 'sense_class' ]=sense_class

    return df18
```



	text	sentiment_score	sense_class
0	merry christmas	0.000000	Neutral
1	witch hunt	0.000000	Neutral
2	china talk going well	0.000000	Neutral
3	america first	0.250000	Positive
4	america first	0.250000	Positive
...	...	...	...
3480	utility cutting rate cite benefit trump tax re...	-0.600000	Negative
3481	today great honor sign new executive order ens...	0.262273	Positive
3482	leaving florida washington dc today pm much wo...	0.378788	Positive

**Figure 4.14. Output of Classification Sense Class**

#### 4.4.3 Wordnet

WordNet is a large lexical database of English which created by Princeton University. WordNet divided word to categories like verbs, noun, adjective and adverbs. It contains 155,287 English words, 175979 synsets and 207016 word-sense pairs.

#### 4.4.4 Sentiwordnet

This is integrated into NLTK as well. It is used for the opinion mining. This helps to deduce the information on polarity from the given example of the problem. SentiWordNet extends wordnet, a lexical word database developed at Princeton (the relationship between words, hence the term net), which is part of the NLTK corpus [19, 20].

## **5. PROGRAMMING LANGUAGE AND SOFTWARE PACKAGE**

### **5.1 PYTHON PROGRAMMING LANGUAGE**

Python is a programming language developed by Guido Van Rossum in Amsterdam at the beginning of 1991-1996. Python got its name from the show performed by the comedy group MontyPython, which developer Rossum loves. Python, unlike most languages, can be run without compiling. It supports Object Oriented Programming (OOP) but there is no obligation to open Class. Python is a programming language that is easy to learn and easily readable. It has cross-platform support and has a place in many different platforms. Python programming language is used in many areas such as Web Applications, Desktop Interface Applications, Game Development, Network Programming, In Scientific, Numerical and Academic Fields, Data analysis and data processing, System Management, Machine Learning, Artificial intelligence, Security, Database. Advantages of Python Programming Language:

- Python is an easy language to learn and develop.
- Python is easier to learn, readable and fast improvements can be made with a simple syntax compared to other programming languages.
- Python is an open source language.
- Python is freely available and distributable for commercial use.
- Python has a large library infrastructure and thanks to these libraries, the development process is very fast.
- The Python language does not need an extra compiler.
- Python IDLE comes automatically, so we don't need a separate compiler to make improvements with Python [22].

## 5.2 PACKAGE TOOLS

### 5.2.1 Anaconda

Anaconda is a platform which includes python distribution, it's a large platform for data scientist who want to use python programming language. In addition to frequently used libraries in data science, artificial intelligence, etc., it also includes tools such as jupyter notebook and spyder. We can install the program by downloading the appropriate version for your operating system from anaconda.org. When you install anaconda, your system will have python, jupyter notebook and spyder installed [23].

### 5.2.2 Jupyterlab

Jupyterlab can edit popular file formats such as, Markdown, JSON, CSV, Vega with live previews. Jupyterlab is also simple to use and can perform many operations in a highly interactive structure with the system in the same project without making tabs and developer see and update live.

Installing Jupyterlab with Conda

```
Conda install -c conda-forge jupyterlab
```

Installing with pip

```
pip install jupyterlab
```

Installing Conda environment to Jupyterlab [24].

### 5.2.3 Python Libraries

Developers attracted created new libraries for Python because of its simplicity. According to creation of new libraries for Python the platform become popular in machine learning experts. There are too many libraries available for users like Pandas, Numpy, SciKit-Learn , Keras, Tensorflow, Scrapy, Nltk, Matplotlib, Scipy, Seaborn, Django, Requests etc. The libraries which were used for the thesis are as follows:

- **Numpy:** is a library for advanced mathematical operations that helps us work with multidimensional arrays and matrices. Numpy library was written to perform statistical operations and data related mathematical operations more effectively. This interface can be utilized for expressing images, sound waves, and other binary raw streams as an array of real numbers in N-dimensional.

- **Pandas:** is a Python package designed to work simply and intuitively with tagged and relational data. It is an excellent tool for preprocessing and analyzing data. It is designed for fast and easy data processing, data management and visualization. Features of Pandas are Re-indexing, Iteration, Sorting, Aggregations, Concatenations.
- **SciKit-Learn:** is a library created by combination of NumPy and SciPy. Scikit-learn is a Python module that combines a wide range of cutting-edge machine learning algorithms for supervised and unsupervised problems. Its useful for extracting features from text and picture.
- **Nltk:** is a libraires which used for Natural Language Processing. NLTK has been prepared to facilitate the teaching and research of NLP and Linguistics, Cognitive Science, Artificial Intelligence related fields. It can get rid of the NLTK's encryption features: text labeling, classification, code names identification, semantic reasoning, etc. It enables complex analysis such as sentiment analysis or automatic summarization.
- **Matplotlib:** Matplotlib is a library for obtaining two dimensional graphic drawings. It produces quality outputs (graphics) in various formats or interactive environments. But it is a low level library for advanced visualization. Matplotlib can be used in Python scripts, Python and IPython shell, Jupyter notebook and web application servers, user graphical interface tools [25, 26].

## 6. MACHINE LEARNING ALGORITHM

In the Twitter sentiment analysis, texts are divided into classes according to polarities, positive, negative and neutral. Classification algorithms essentially divide the incoming document, text or instance into 2 or more categories. Classification is a Supervised Learning algorithm and it based on the predefined training data. We will examine the binary classification algorithm, especially in twitter sentiment analysis, because tweets are tagged as negative or positive. The algorithms used in this thesis are Support Vector Machine, Logistic Regression, Naïve Bayes, Random Forest and XGboost.

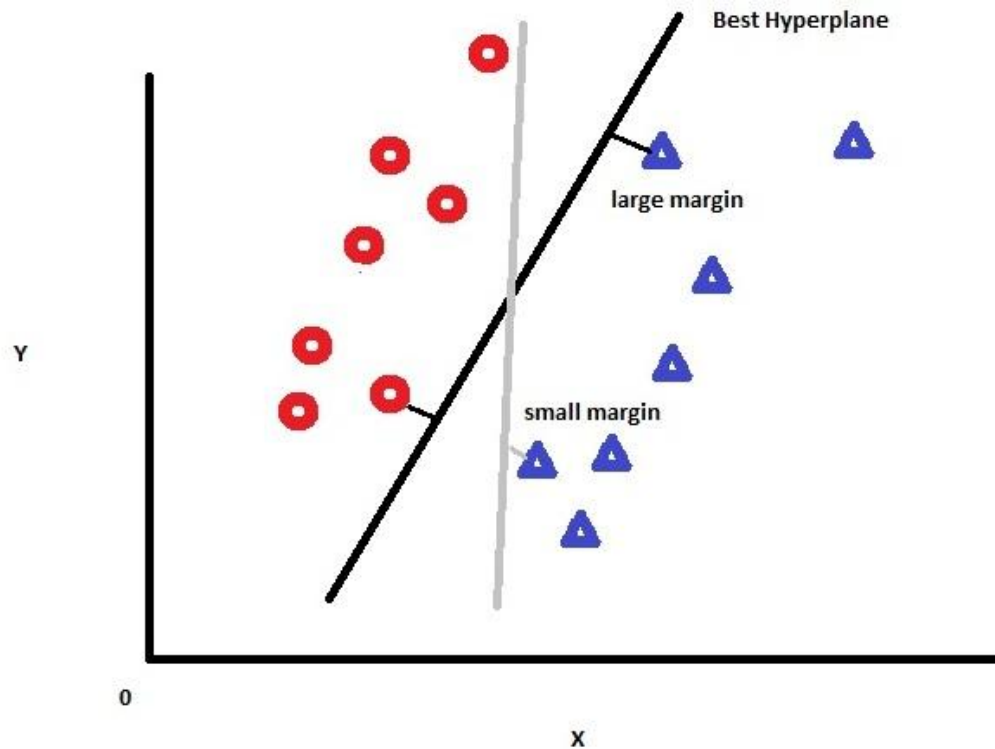
### 6.1 SUPPORT VECTOR MACHINE

Support Vector Machine is like a logistic regression, but more advanced, classification algorithm. SVM's goal is to find a hyperplane that allows the two classes to be optimally separated. The algorithm allows two classes to adjust the line to be drawn so that it passes the furthest place to its components.

Using machine learning, the information sets are examined to point out a relationship. The relationships are then placed along the X/Y axis, with a line running through them to predict further relationships.

In this thesis the output is Y value which is classified as a Negative and Positive according to sentiment analysis (polarity). The tweets text (word and phrase) as input value X(feature).

SVM use algorithm to train and classify text data according to sentiment polarity, to move one step further from X/Y prediction. We'll train classifier to output an an X/Y coordinate as either Negative or Positive. SVM assign a hyperplane seperates Negative and Positive tags from each other. On the 2-dimension it is a line seperates the Negative as one side and the Positive the other side. To maximize the SVM select the hyperplane which is large distance between the two tags shown in Figure 6.1.



**Figure 6.1. Support Vector Machine Hyperplane**

Some conditions the data are more complex and its difficult to separate by a single line. The “Kernel Trick” use for solve this problem. Adding a 3.rd dimension with Z coordinate can help the separate the tags from each other [27].

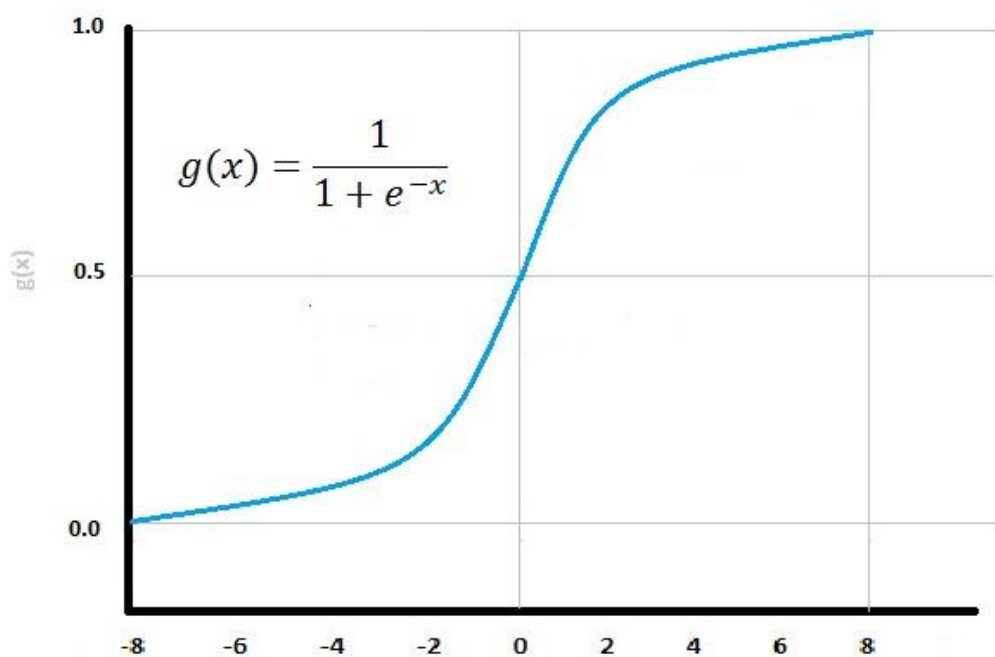
## 6.2 LOGISTIC REGRESSION

Logistic regression is a statistical method used to analyze a data set with one or more independent variables that determine a result. The result which named also dependent variable contains binary value like 0 and 1 or True or False. In twitter data sentiment analysis the dependent variable tagged as Negative or Positive. The aim of Logistic Regression is to find the suitable model for dependent variable and independent variable. Logistic regression produces the coefficients, standard errors, and significance levels of a formula to predict the probability of the existence of the characteristics of interest, the logit transformation

$$\text{logit}(p) = b_0 + b_1x_1 + b_2x_2 + \dots + b_kx_k$$

$$\text{odds} = \frac{p}{1-p} = \frac{\text{probability of presence of characteristic}}{\text{probability of absence characteristic}} \quad (6.1)$$

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right)$$



**Figure 6.2. Logistic Regression S-Shape Curve**

The Logistic Regression take the name from the logit function which is named also sigmoid function shown in Figure 6.2. Function has a curve like s-shape and range between 0 and 1. The values between 0 and 1 transform to 0 or 1 according to threshold classifier [28].

### 6.3 NAÏVE BAYES

Naïve Bayes is Machine Learning algorithm based on Bayes probability theorem.

A Naïve Bayes aim is to calculate the probability of a particular sample belonging to each class on a conditional probability basis. Naïve Bayes used for classification problems.

Lazy learner algorithm and it also works with unbalanced datasets. If a data in the test set does not have a corresponding value in the train set, it is called Zero frequency and the solution is provided with the Laplace method.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (6.3)$$

$P(A|B)$ : The probability of event A occurring when event B occurs

$P(B|A)$ : Probability of B occurrence when event A occurs

$P(A)$ : Probability of event A occurring

$P(B)$ : Probability of event B occurring

There is three kind of Naïve Bayes model:

- **Gauss Naïve Bayes:** Feature has a normal distribution and it used for classification
- **Bernoulli Naïve Bayes:** This model used when a feature vectors have binary. The parameters we use to predict the class variable take only yes or no values, for example, whether there is a word in the text or not.
- **Multinomial Naïve Bayes:** Used for text classification. The properties / predictors used by the classifier are the words found in the document. is the frequency. In this thesis Multinomial Naïve Bayes model is used for prediction accuracy in sentiment analysis of Twitter text data.

Naive Bayes are used in text classification, spam filtering, sentiment analysis and suggestion systems [29, 30].



## 6.4 RANDOM FOREST

Random Forest is based on the evaluation of the predictions produced by more than one decision tree. Observations for trees are selected by bootstrap random sample selection method and variables by random subspace method.

- In each node of the decision tree, the best branching (Information Gain) variable is chosen from a smaller number of variables randomly selected among all variables.
- 2/3 of the data set is used in creating trees. The external data is used for performance evaluation of trees and determination of variable importance.
- Random variable selection is made at each node. As much as the square root of  $p$  for the classification process
- For the final estimation, the estimation values are requested from the trees, taking into account the previously calculated error rates of each tree [31].

## 6.5 EXTREME GRADIENT BOOSTING

XGBoost has been optimized to increase the speed and prediction performance of the Gradient Boosting Machine; It is scalable and can be integrated into different platforms. (Tianqe Chen, 2014)

- XGBoost can used with R, Python, Hadoop, Scala, Julia.
- Scalable (Predictive function and system integration)
- Fast
- Prediction success is high.
- It has proven its success in many Kaggle competitions.
- By allowing In Memory to work on Apache Spark, XGBoost can work with data cached on the Ram, so there is no need for disk read-write [32].

## 6.6 ACCURACY

Accuracy is a method used to compare the success of classification algorithm. Accuracy is the ratio of the total number of correctly classified data to the total number of data. Accuracy value is between 0 and 1. 0 is the worst score and 1 is the best score for accuracy. Accuracy can be calculated by the equation given in (6.4)

$$Accuracy = \frac{True\ Negative + True\ Positive}{True\ Positive + False\ Positive + True\ Negative + False\ Negative} \quad (6.4)$$

where the confusion matrix shown in Table 6.1.

**Table 6.1. Confusion Matrix**

Prediction \ Real	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative(FN)	True Negative(TN)

## 7. METHODOLOGY

### 7.1 PREPROCESSING

In the preprocessing phase, first performed the process of adding our data set which name is "Training set.csv" file containing all of the Donald Trump tweets. Tweets data include from tweets posted since 2009-05-04 to 2020-11-06. Tweet data has 8 feature and 55090 tweets.

Tweets imported to dataframe which called df according to data structure shown in Table 7.1. For performing sentiment analysis and feature extraction the most important data is text data. Before sentiment analysis and feature extraction the preprocessing step provide to clean the text data from noisy data or unnecessary object. After whole preprocessing steps the clean text data shown in Figure 7.1.

**Table 7.1. Import Data and Set into DataFrame**

```
# Import package
import pandas as pd

data = pd.read_csv("Training sets.csv")

# Build DataFrame of tweet texts and languages
df = pd.DataFrame(data, columns=['id', 'text', 'isRetweet', 'isDeleted', 'device', 'favorites',
'retweets', 'date'])

# Print head of DataFrame
print(df.head())
```

```

      id                                     text \
0    98454970654916608 Republicans and Democrats have both created ou...
1   1234653427789070336 I was thrilled to be back in the Great city of...
2   1218010753434820614 RT @CBS_Herridge: READ: Letter to surveillance...
3   1304875170860015617 The Unsolicited Mail In Ballot Scam is a major...
4   1218159531554897920 RT @MZHemingway: Very friendly telling of even...

  isRetweet isDeleted      device  favorites  retweets \
0          f         f      TweetDeck      49      255
1          f         f  Twitter for iPhone  73748   17404
2          t         f  Twitter for iPhone      0     7396
3          f         f  Twitter for iPhone  80527   23502
4          t         f  Twitter for iPhone      0     9081

      date
0  2011-08-02 18:07:48
1  2020-03-03 01:34:50
2  2020-01-17 03:22:47
3  2020-09-12 20:10:58
4  2020-01-17 13:13:59

```

**Figure 7.1. Output of the Twitter Dataset to Dataframe**

In this thesis, in addition to analyzing all tweet data since 2009 to 2020, the data for the years 2017-2018-2019-2020 covering the presidency of President Donald Trump are analyzed separately and whole. Sentiment analysis, feature extraction and classification of sentiment analysis by machine learning algorithms have been made according to specified years. Splitting the data by Python Code according to years 2017-2018-2019-2020 shown in Table 7.2. `df.info()` function in Python programming language shows the entities, columns, data type and memory usage information of dataframe which named as `df`. shown in Table 7.3.

**Table 7.2. Splitting Data to According to Years of 2017-2018-2019-2020**

```

year2017= (df['date'] >='2017-01-01') & (df['date'] <='2017-12-31')
df17= df[year2017].copy()
df17.info()

year2018= (df['date'] >='2018-01-01') & (df['date'] <='2018-12-31')
df18= df[year2018].copy()
df18.info()

year2019= (df['date'] >='2019-01-01') & (df['date'] <='2019-12-31')
df19= df[year2019].copy()
df19.info()

year2020= (df['date'] >='2020-01-01') & (df['date'] <='2020-11-06')
df20= df[year2020].copy()
df20.info()

```

**Table 7.3. Dataframe Information**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55090 entries, 0 to 55089
Data columns (total 8 columns):
#   Column      Non-Null Count  Dtype
---  -
0   id           55090 non-null  int64
1   text        55090 non-null  object
2   isRetweet   55090 non-null  object
3   isDeleted   55090 non-null  object
4   device      55090 non-null  object
5   favorites   55090 non-null  int64
6   retweets    55090 non-null  int64
7   date        55090 non-null  object
dtypes: int64(3), object(5)
memory usage: 3.4+ MB
```

In preprocessing steps include remove tag, hashtag, RT, url, number, punctuation, non-English characters. After removing unnecessary structure, lowercase applied to text data. The related libraries in Python programming language are imported such as pandas, numpy, re (regex: regular expression). The pre-defined cleanTxt and Lambda function used for preprocessing steps shown in Table 7.4. The output of after preprocessing shown in Figure 7.2.

**Table 7.4. Preprocessing Text Data**

```
import pandas as pd
import numpy as np
import re

def cleanTxt(txt):
    txt = re.sub('@[A-Za-z0-9]+', '', txt) #Removing @mentions
    txt = re.sub('#', '', txt) # Removing '#' hash tag
    txt = re.sub(r'[\w\s]', '', txt)
    txt = re.sub('_', '', txt)
    txt = re.sub('amp', '', txt)
    txt = re.sub('[^\x00-\x7F]+', '', txt) #Removing non-English characters
    txt = re.sub('RT[\s]+', '', txt) # Removing RT
    txt = re.sub('https?:\s+/', '', txt) # Removing hyperlink
    return txt

# Clean the tweets
df['text'] = df['text'].apply(cleanTxt)

#lowercase
df['text'] = df['text'].apply(lambda x: " ".join(x.lower() for x in x.split()))
```

```

df.head()
#remove punctuations
df['text'] = df['text'].str.replace(r'^\w\s|', '')
df.head()
#remove number
df['text'] = df['text'].str.replace('\d', '')
df.head()
#remove url link
df['text'] = df['text'].str.replace('http\S+ | www.\S+', '', case=False)
df.head()

```

	id	text	isRetweet	isDeleted	device	favorites	retweets	date
0	98454970654916608	republicans and democrats have both created ou...	f	f	TweetDeck	49	255	2011-08-02 18:07:48
1	1234653427789070336	i was thrilled to be back in the great city of...	f	f	Twitter for iPhone	73748	17404	2020-03-03 01:34:50
2	1218010753434820614	herridge read letter to surveillance court obt...	t	f	Twitter for iPhone	0	7396	2020-01-17 03:22:47
3	1304875170860015617	the unsolicited mail in ballot scam is a major...	f	f	Twitter for iPhone	80527	23502	2020-09-12 20:10:58
4	1218159531554897920	very friendly telling of events here about com...	t	f	Twitter for iPhone	0	9081	2020-01-17 13:13:59

**Figure 7.2. Output of the Twitter Data After Preprocessing**

The another preprocessing steps are cleaning the tweet text data from stopwords like “a,an,” and lemmatization of text data. In this step the nltk and textblob library used for download stopwords and wordnet lexicon. The code used for remove stopwords and lemmatization shown in Table 7.5

**Table 7.5. Delete Stop Words and Lemmitization Process**

```

#stopwords
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
sw = stopwords.words('english')
df['text'] = df['text'].apply(lambda x: " ".join(x for x in x.split() if x not in sw))

import nltk
nltk.download('wordnet')
#lemmi
!pip install textblob
from textblob import Word
#nltk.download('wordnet')
df['text'] = df['text'].apply(lambda x: " ".join([Word(word).lemmatize() for word in x.split()]))

```

After preprocessing the tweets data the number of entities decrease. The cleaned data shown in Table 7.6 according to years, Presidency Term (2017-2020) and all data.

**Table 7.6. Number of Data Before and After Preprocessing**

Data	Before Cleaning	After Cleaning
Tweets 2017	2593	2587
Tweets 2018	3556	3485
Tweets 2019	7783	7428
Tweets 2020	10911	9940
Tweets 2017-2020	24843	23440
All Tweets	55090	53609

## 7.2 FREQUENCY OF WORDS IN DATA AND VISULIZATION

In this section WordCloud, word count (frequency), and the device which used to post the tweets and their visualization according to years from 2017 to 2020, Presidency Term (2017-2020) and all data are available. The all cleaned tweet data is set in a text file. The text file is used for WordCloud which provides an aesthetic representation of the words according to frequency of word. If the count of word is higher, the character font of the word bigger than other word in WordCloud. The output of the all text data shown in Figure 7.3.

```
text = " ".join(i for i in df.text)
text

'republican democrat created economic problem thrilled back great
n patriot love country cherish value respect law always put amer:
urveillance court obtained cbs news question disciplinary action
ocrat know almost recent election using system even though much :
issing ballot fraud friendly telling event comeys apparent leakin
toric step protect constitutional right pray public school im rur
worse sustainable county china u getting little exercise morning
```

**Figure 7.3. Text Data of All Tweets**

The related Python library are installed such as wordcloud, matplotlib and related configuration 'background, interpolation, axis' made in the code section shown in Table 7.7







Figure 7.5. WordCloud of Tweets 2018



Figure 7.6. WordCloud of Tweets 2019

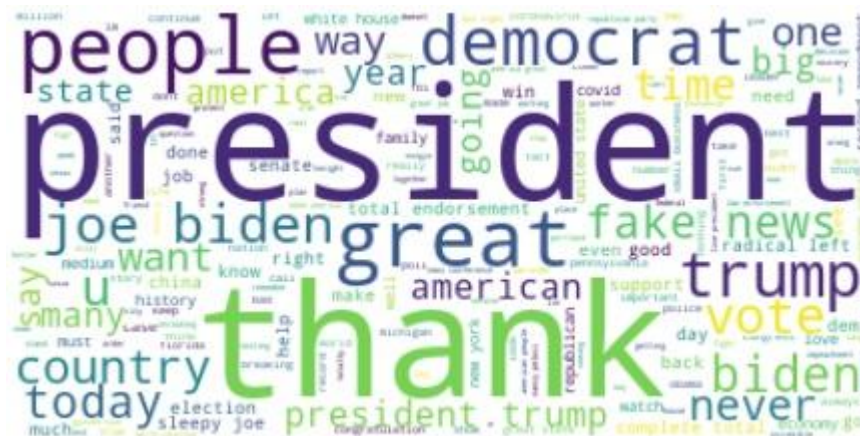


Figure 7.7. WordCloud of Tweets 2020



**Figure 7.8. WordCloud of 2017-2020 Presidency Term**



**Figure 7.9. WordCloud of All Data**

Python code word frequency of the tweets are listed by according to years and whole data shown Table 7.8 and the first top ten words and count of words shown in Figure 7.10. The Plot Bar Graph of first top twenty words and count of words shown in Figure 7.11, Figure 7.12 and Figure 7.13.

**Table 7.8. Python Code of Word Frequency**

```
df=(df['text'].str.split(expand=True)
    .stack()
    .value_counts()
    .rename_axis('text')
    .reset_index(name='count'))
df.head(10)
```

### Word Frequency Tweets According to Years, Presidency Term and Whole Data

	text	count	text	count	text	count	text	count	text	count	text	count
0	great	491	great	877	great	1160	president	1204	great	3728	trump	7529
1	u	214	people	468	president	1121	great	1200	president	2854	great	7462
2	people	210	country	424	democrat	1002	trump	947	democrat	2202	president	4684
3	news	208	u	380	trump	716	biden	790	trump	2169	thank	3526
4	job	184	president	370	people	630	people	763	people	2071	people	3382
5	fake	184	border	368	country	586	democrat	738	country	1611	u	3104
6	tax	180	trump	354	u	493	thank	695	thank	1611	country	2680
7	thank	178	democrat	354	year	490	american	638	u	1548	get	2558
8	today	168	state	310	state	489	joe	631	state	1510	new	2485
9	america	168	job	294	thank	485	state	591	news	1494	america	2403

2017

2018

2019

2020

2017-2020

2009-2020

**Figure 7.10. Word Frequency of Tweets**

As shown in Figure 7.4 first year of Donald Trump Presidency (2017), Figure 7.7 last year of Presidency and election term (2020), ‘great’, ‘thank’, ‘people’ ‘u’ words used in 2017, ‘president’, ‘thank’, ‘great’, ‘people’, ‘democrat’ ‘Joe Biden’ words used in 2020. The words President Donald Trump uses in his tweets appear to be related to the agenda and current issues.

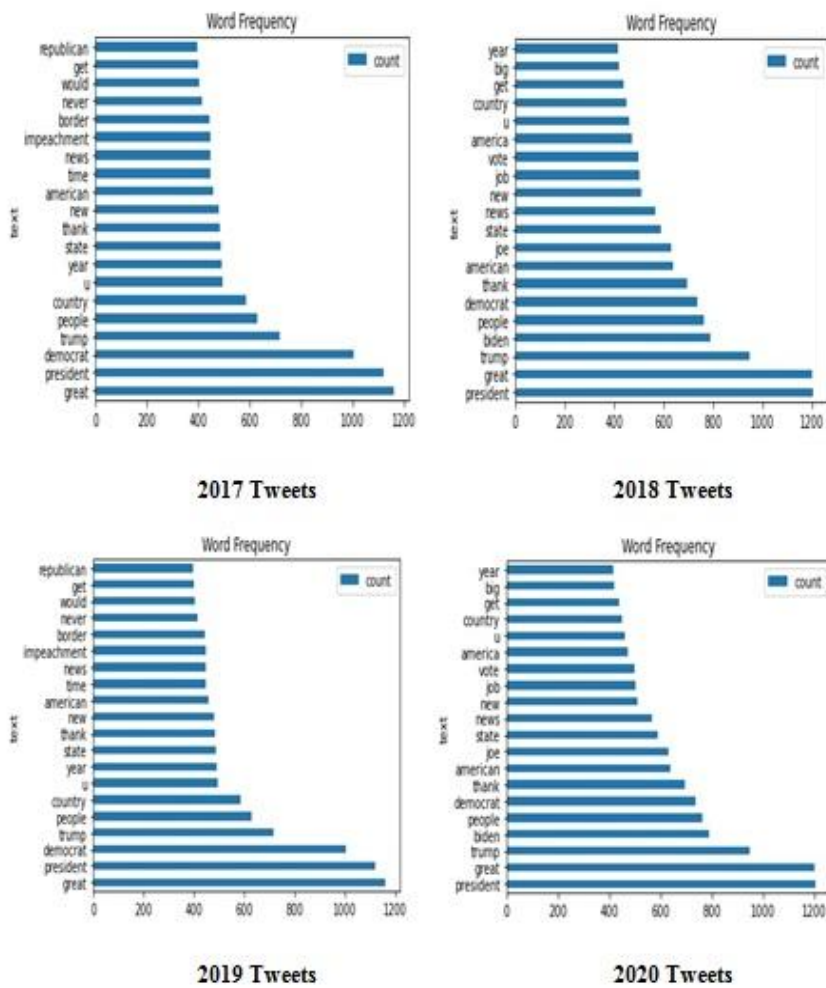


Figure 7.11. Plot Bar Graph Word Frequency Years 2017-2018-2019-2020

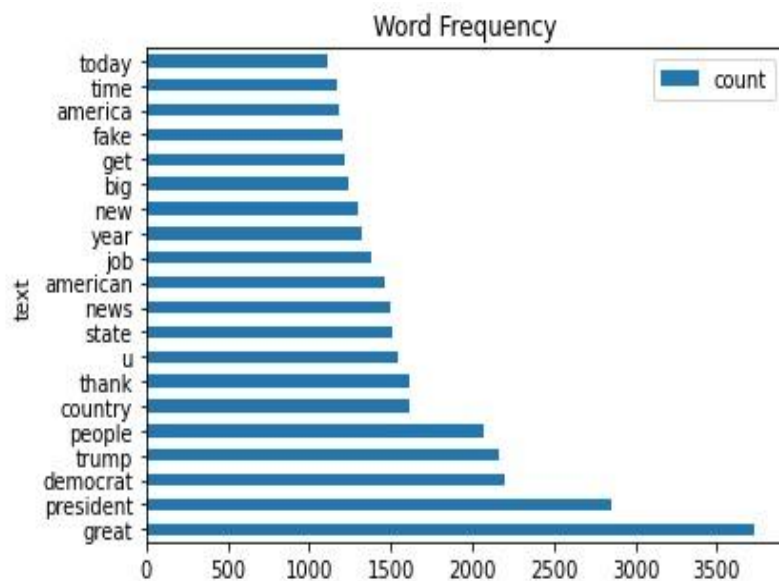
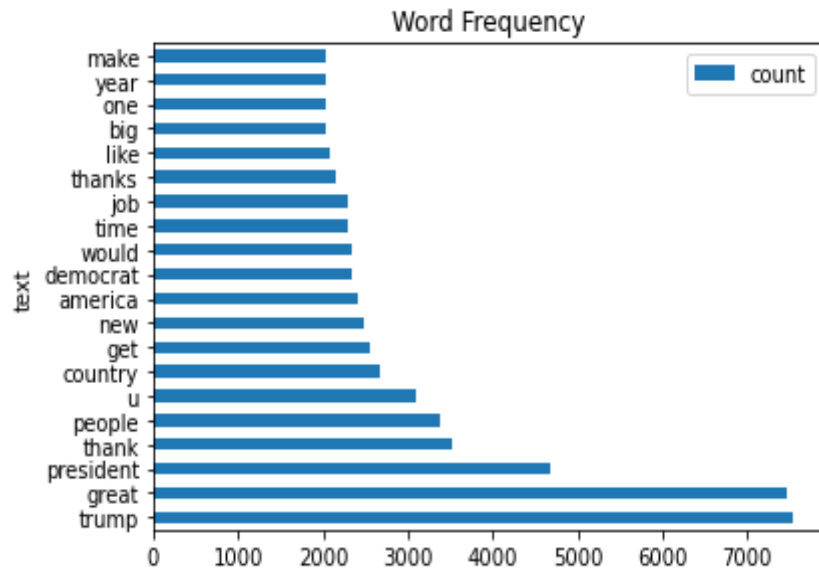


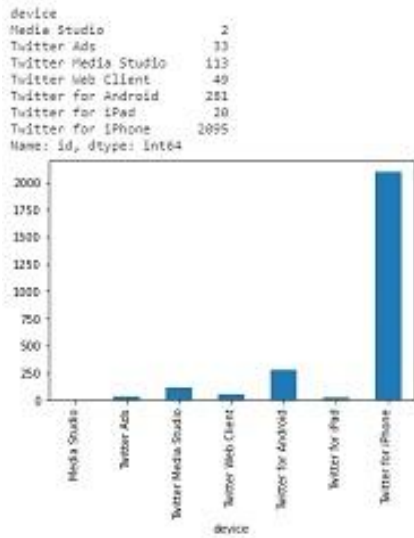
Figure 7.12. Plot Bar Graph Word Frequency 2017-2020



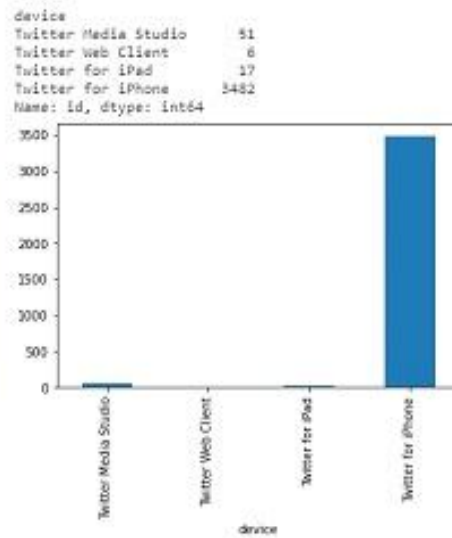
**Figure 7.13. Plot Bar Graph Word Frequency of All Data**

The device column of the tweet dataframe is used for the device which used to post the tweets from different source shown in Figure 7.14, Figure 7.15 and Figure 7.16. Looking at the tweets sent by years, it is seen that the tweets are mostly sent from iPhone. Looking at the tweets sent by years, it is seen that the tweets are mostly sent from iPhone. Considering all data, it is seen that iPhone, Android and Web client are used respectively.

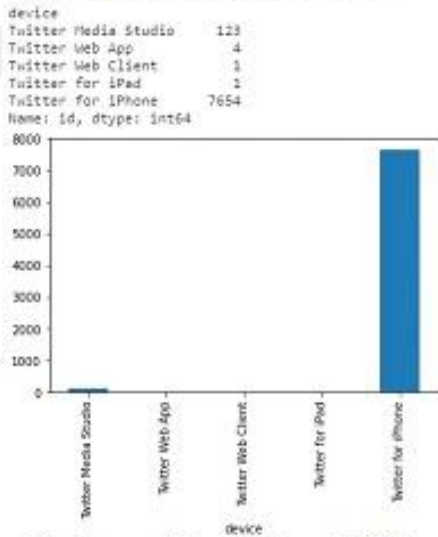




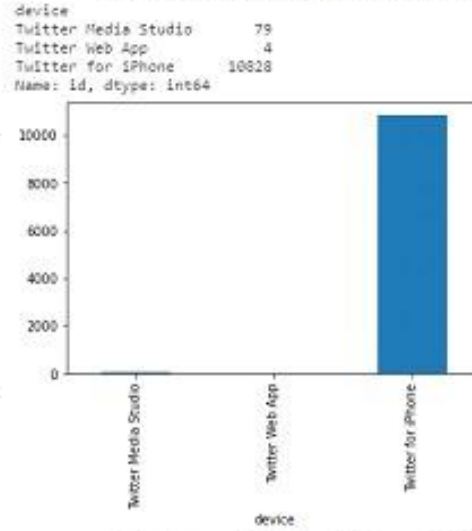
**Device used to post Tweets 2017**



**Device used to Post Tweets 2018**



**Device used to post Tweets 2019**



**Device used to Post Tweets 2020**

**Figure 7.14. Device Used to Post Years 2017-2018-2019-2020 Tweets**

```

device
Twitter Media Studio      79
Twitter Web App          4
Twitter for iPhone      10828
Name: id, dtype: int64

```

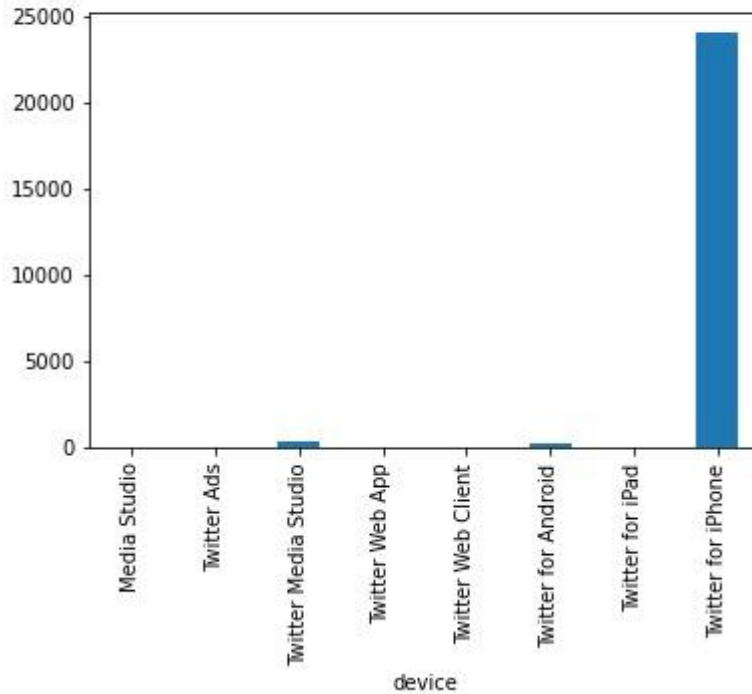


Figure 7.15. Device Used to Post 2017-2020 Tweets

```

device
Facebook                105
Instagram               133
Media Studio            2
Neatly For BlackBerry 10  5
Periscope               7
TweetDeck              482
TwitLonger Beta        405
Twitlonger             23
Twitter Ads            97
Twitter Media Studio   368
Twitter Mirror for iPad 1
Twitter QandA          10
Twitter Web App        64
Twitter Web Client    12182
Twitter for Android   14545
Twitter for BlackBerry 97
Twitter for Websites  1
Twitter for iPad       60
Twitter for iPhone    26493
Vine - Make a Scene    10
Name: id, dtype: int64

```

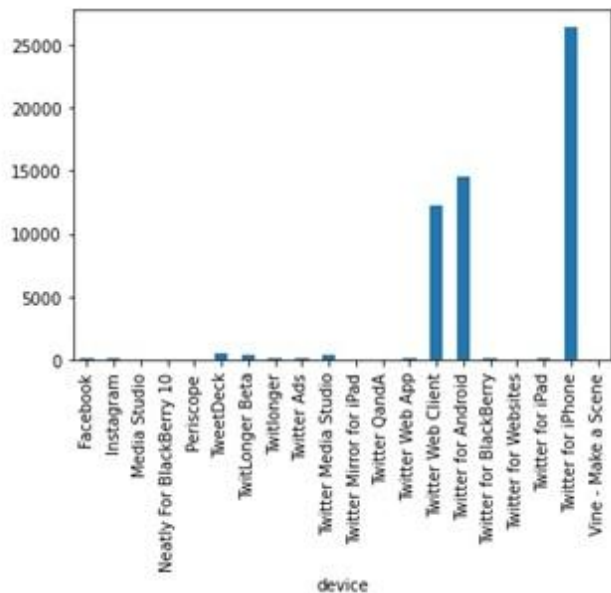


Figure 7.16. Device Used to Post All Tweets

### 7.3 SENTIMENT ANALYSIS

In this section after the text preprocessing and reindexing dataframe the TextBlob library used for sentiment analysis and the sentiment polarity calculated for each tweet text and assigned to sentiment\_score. If total sentiment\_score is lower than 0 its sense\_class labeled as Negative , if the sentiment\_score is higher than 0 its sense\_class labeled as Positive and sentiment\_score is equal to zero sense\_class labeled as Neutral. The Python code of the sentiment analysis and output of the analysis shown Table 7.9 and Figure 7.17.

**Table 7.9. Python Code of Sentiment Analysis**

```
from textblob import TextBlob
def sentiment_scoring(df):
    text = df["text"]
    for i in range(0,len(text)):
        textB = TextBlob(text[i])
        sentiment_score = textB.sentiment.polarity
        df.at[i, 'sentiment_score']=sentiment_score
        if sentiment_score <0.00:
            sense_class = 'Negative'
            df.at[i, 'sense_class' ]=sense_class
        elif sentiment_score >0.00:
            sense_class = 'Positive'
            df.at[i, 'sense_class' ]=sense_class
        else:
            sense_class = 'Neutral'
            df.at[i, 'sense_class' ]=sense_class
    return df
sentiment_scoring(df)
```



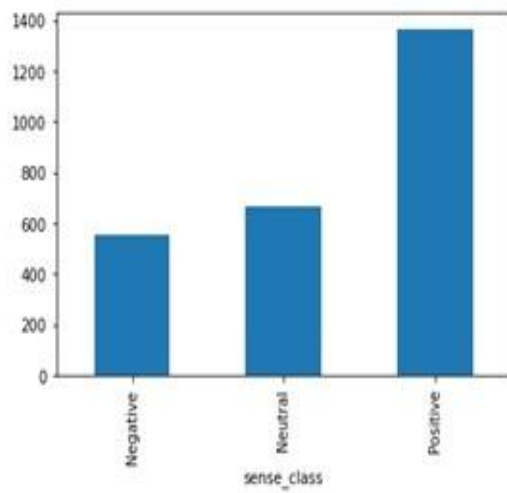
1	1234653427789070336	thrilled back great city charlotte north carol...	f	f	Twitter for iPhone	73748	17404	2020- 03-03 01:34:50	0.450000	Positive
2	1218010753434820614	herridge read letter surveillance court obtain...	t	f	Twitter for iPhone	0	7396	2020- 01-17 03:22:47	0.100000	Positive
3	1304875170860015617	unsolicited mail ballot scam major threat demo...	f	f	Twitter for iPhone	80527	23502	2020- 09-12 20:10:58	0.029464	Positive
4	1218159531554897920	friendly telling event comeys apparent leaking...	t	f	Twitter for iPhone	0	9081	2020- 01-17 13:13:59	0.212500	Positive
...	...	...	...	...	...	...	...	...	...	...
53604	1319485303363571714	dont know think continue lie want ban fracking...	t	f	Twitter for iPhone	0	20683	2020- 10-23 03:46:25	0.000000	Neutral
53605	1319484210101379072	president excels communicating directly americ...	t	f	Twitter for iPhone	0	9869	2020- 10-23 03:42:05	0.000000	Neutral

**Figure 7.17. Output of Sentiment Analysis**

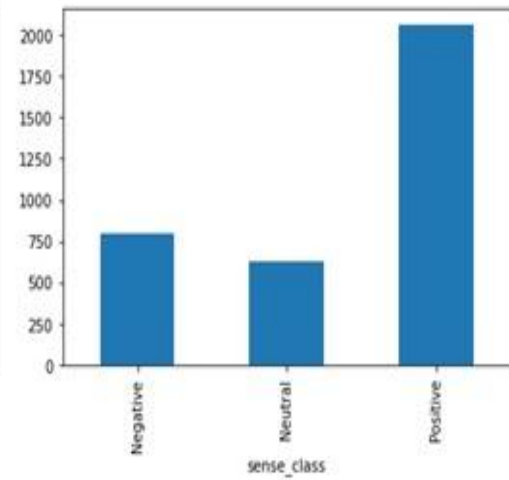
The count and distribution of sentiment class which name as sense\_class of tweet data over the years, Presidency Term (2017-2020) and whole shown Table 7.10, Figure 7.18, Figure 7.19, Figure 7.20. Except for the tweet data for 2018, the sentiment class order is positive, neutral and negative from high to low. The sequence like positive, negative and neutral on Tweets of 2018.

**Table 7.10. Sense Class of Tweets according to Years and Whole Data**

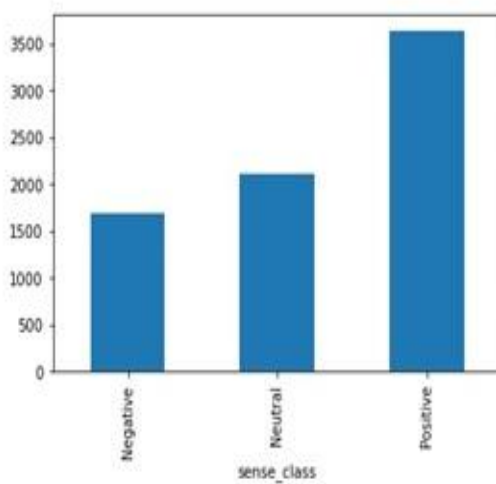
Data	Negative	Neutral	Positive
Tweets 2017	555	666	1366
Tweets 2018	794	630	2061
Tweets 2019	1684	2106	3638
Tweets 2020	2052	3958	4901
Tweets 2017-2020	5053	6589	11798
All Tweets	9795	15713	28101



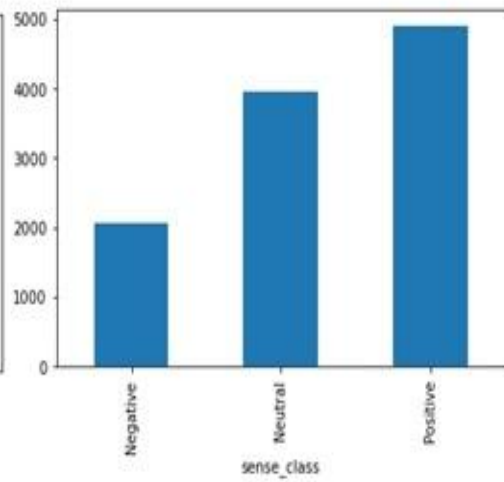
Tweets 2017



Tweets 2018

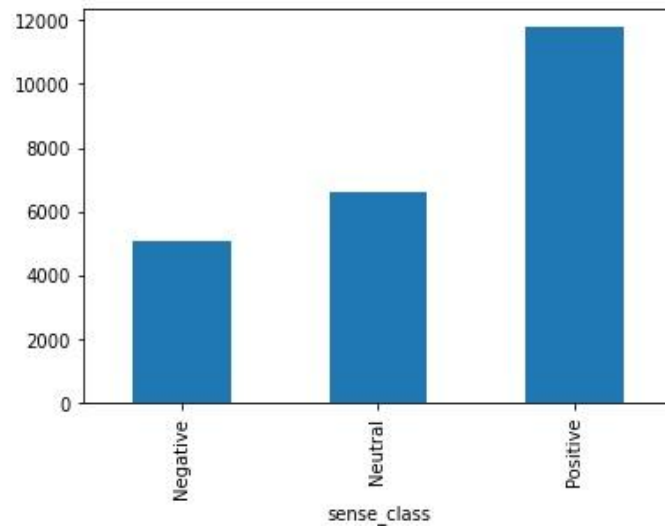


Tweets 2019

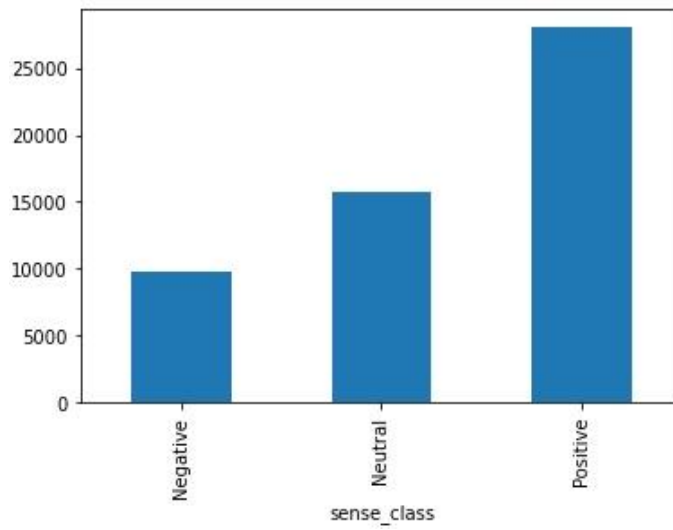


Tweets 2020

**Figure 7.18. Sense Class Tweets Years 2017-2018-2019-2020**



**Figure 7.19. Sense Class of 2017-2020**



**Figure 7.20. Sense Class of All Data**

## 7.4 DATA FEATURE EXTRACTION AND MODELLING

In this section, feature extraction from text data and fitting machine learning model and comparison of accuracy values are performed. The related library such as sklearn, keras and their methods, machine learning algorithm like Support Vector Machine, Logistic Regression, Naïve Bayes, Random Forest, XGBoost are installed.

The preprocessing.LabelEncoder() function was used for converting the value of “sense\_class” string value to numeric value. Encoder was applied for train set and test set at (independent variable) same time. The Feature Extraction such as Count Vector, Tf-idf N-gram Level, Tf-idf Char Level and Tf-idf Word Level applied to the dependent variable train and test set at same time shown in Table 7.11.

Each feature extraction has been examined one by one over the years and by the whole, using Support Vector Machine, Logistic Regression, Naïve Bayes, Random Forest, XGBoost machine learning algorithms shown in Table 7.12. Finally, the performances of the machine learning models were realized in different test\_sizes such as 0.33, 0.40.

**Table 7.11. Feature Extraction and Fitting to Train and Test Set**

```
from textblob import TextBlob
from sklearn import model_selection, preprocessing, linear_model, naive_bayes,
metrics
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn import decomposition, ensemble

import pandas, xgboost, numpy, textblob, string
from keras.preprocessing import text, sequence
from keras import layers, models, optimizers

from warnings import filterwarnings
filterwarnings('ignore')

train_x, test_x, train_y, test_y = model_selection.train_test_split(df["text"],
                                                                    df["sense_class"], random_state = 1)
```

```
encoder = preprocessing.LabelEncoder()

train_y = encoder.fit_transform(train_y)

test_y = encoder.fit_transform(test_y)

from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer()

vectorizer.fit(train_x)

x_traincountv = vectorizer.transform(train_x)

x_testcountv = vectorizer.transform(test_x)

from sklearn.feature_extraction.text import TfidfVectorizer

tword_vectorizer = TfidfVectorizer()

tword_vectorizer.fit(train_x)

x_train_tword = tword_vectorizer.transform(train_x)

x_test_tword = tword_vectorizer.transform(test_x)

tgram_vectorizer = TfidfVectorizer(ngram_range = (1,2))

tgram_vectorizer.fit(train_x)
```

```

x_train_tngram = tngram_vectorizer.transform(train_x)

x_test_tngram = tfidf_ngram_vectorizer.transform(test_x)

tchars_vectorizer = TfidfVectorizer(analyzer = "char", ngram_range = (1,5))

tchars_vectorizer.fit(train_x)

x_train_tchars = tfidf_chars_vectorizer.transform(train_x)
x_test_tchars = tfidf_chars_vectorizer.transform(test_x)

```

**Table 7.12. Python Code of Machine Learning Algorithm Accuracy**

## **SUPPORT VECTOR MACHINE**

### **– Count Vector**

```

from sklearn.svm import SVC

Svm = SVC(kernel="linear")

Supportvector_model = Svm.fit(x_traincountv, train_y)

accuracy = model_selection.cross_val_score(Supportvector_model,

                                            x_testcountv,

                                            test_y, cv=10).mean()

print("Count Vectors Accuracy Score:", accuracy)

```

– **Tf-Idf Word Level**

```
Svm = SVC(kernel="linear")

Supportvector_model = Svm.fit(x_train_tword, train_y)

accuracy = model_selection.cross_val_score(Supportvector_model,

                                           x_test_tword,

                                           test_y, cv=10).mean()

print("Word-Level TF-IDF Accuracy Score:", accuracy)
```

– **Tf-Idf N-gram Level**

```
Svm = SVC(kernel="linear")

Supportvector_model = Svm.fit(x_train_tngram, train_y)

accuracy = model_selection.cross_val_score(Supportvector_model,

                                           x_test_tngram,

                                           test_y, cv=10).mean()

print("N-GRAM TF-IDF Accuracy Score:", accuracy)
```

– **Tf-Idf Char Level**

```
Svm = SVC(kernel="linear")

Supportvector_model = Svm.fit(x_train_tchars, train_y)

accuracy = model_selection.cross_val_score(Supportvector_model,

                                           x_test_tchars,

                                           test_y, cv=10).mean()

print("CHARLEVEL Accuracy Score:", accuracy)
```

## LOGISTIC REGRESSION

### – Count Vector

```
from sklearn import linear_model

log = linear_model.LogisticRegression(max_iter=1000)

log_model = log.fit(x_traincountv, train_y)

accuracy = model_selection.cross_val_score(log_model,

                                             x_testcountv,

                                             test_y, cv=10).mean()

print("Count Vectors Accuracy Score:", accuracy)
```

### – Tf-Idf Word Level

```
log = linear_model.LogisticRegression(max_iter=1000)

log_model = log.fit(x_train_tword, train_y)

accuracy = model_selection.cross_val_score(log_model,

                                             x_test_tword,

                                             test_y,

                                             cv = 10).mean()

print("Word-Level TF-IDF Accuracy Score:", accuracy)
```

### – Tf-Idf N-gram Level

```
log = linear_model.LogisticRegression(max_iter=1000)
```



```
log_model = log.fit(x_train_tngram,train_y)

accuracy = model_selection.cross_val_score(log_model,

                                           x_test_tngram,

                                           test_y,

                                           cv = 10).mean()

print("N-GRAM TF-IDF Accuracy Score:", accuracy)
```

#### – Tf-Idf Char Level

```
log = linear_model.LogisticRegression(max_iter=1000)

log_model = log.fit(x_train_tchars,train_y)

accuracy = model_selection.cross_val_score(log_model,

                                           x_test_tchars,

                                           test_y,

                                           cv = 10).mean()

print("CHARLEVEL Accuracy Score:", accuracy)
```

### NAIVE BAYES

#### – Count Vector

```
nb = naive_bayes.MultinomialNB()

naiveb_model = nb.fit(x_traincountv,train_y)

accuracy = model_selection.cross_val_score(naiveb_model,
```

```
x_testcountv,  
  
test_y,  
  
cv = 10).mean()  
  
print("Count Vectors Accuracy Score:", accuracy)
```

#### – Tf-Idf Word Level

```
nb = naive_bayes.MultinomialNB()  
  
naiveb_model = nb.fit(x_train_tword,train_y)  
  
accuracy = model_selection.cross_val_score(naiveb_model,  
  
x_test_tword,  
  
test_y,  
  
cv = 10).mean()  
  
print("Word-Level TF-IDF Accuracy Score:", accuracy)
```

#### – Tf-Idf N-gram Level

```
nb = naive_bayes.MultinomialNB()  
  
naiveb_model = nb.fit(x_train_tfidf_ngram,train_y)  
  
accuracy = model_selection.cross_val_score(naiveb_model,  
  
x_test_tfidf_ngram,  
  
test_y,  
  
cv = 10).mean()  
  
print("N-GRAM TF-IDF Accuracy Score:", accuracy)
```

– **Tf-Idf Char Level**

```
nb = naive_bayes.MultinomialNB()

naiveb_model = nb.fit(x_train_tchars,train_y)

accuracy = model_selection.cross_val_score(naiveb_model,

                                           x_test_tchars,

                                           test_y,

                                           cv = 10).mean()

print("CHARLEVEL Accuracy Score:", accuracy)
```

**RANDOM FOREST**

– **Count Vector**

```
rf = ensemble.RandomForestClassifier()

randomf_model = rf.fit(x_traincountv,train_y)

accuracy = model_selection.cross_val_score(randomf_model,

                                           x_testcountv,

                                           test_y,

                                           cv = 10).mean()

print("Count Vectors Accuracy Score:", accuracy)
```

– **Tf-Idf Word Level**

```
rf = ensemble.RandomForestClassifier()

randomf_model = rf.fit(x_train_tword,train_y)

accuracy = model_selection.cross_val_score(randomf_model,
```

```
x_test_tword,  
  
test_y,  
  
cv = 10).mean()  
  
print("Word-Level TF-IDF Accuracy Score:", accuracy)
```

#### – Tf-Idf N-gram Level

```
rf = ensemble.RandomForestClassifier()  
  
randomf_model = rf.fit(x_train_tngram,train_y)  
  
accuracy = model_selection.cross_val_score(randomf_model,  
  
x_test_tngram,  
  
test_y,  
  
cv = 10).mean()  
  
print("N-GRAM TF-IDF Accuracy Score:", accuracy)
```

#### – Tf-Idf Char Level

```
rf = ensemble.RandomForestClassifier()  
  
randomf_model = rf.fit(x_train_tchars,train_y)  
  
accuracy = model_selection.cross_val_score(randomf_model,  
  
x_test_tchars,  
  
test_y,  
  
cv = 10).mean()  
  
print("CHARLEVEL Accuracy Score:", accuracy)
```

## **XGBOOST**

### **– Count Vector**

```
xgb = xgboost.XGBClassifier()

xgboost_model = xgb.fit(x_traincountv,train_y)

accuracy = model_selection.cross_val_score(xgboost_model,

                                           x_testcountv,

                                           test_y,

                                           cv = 10).mean()

print("Count Vectors Accuracy Score:", accuracy)
```

### **– Tf-Idf Word Level**

```
xgb = xgboost.XGBClassifier()

xgboost_model = xgb.fit(x_train_tword,train_y)

accuracy = model_selection.cross_val_score(xgboost_model,

                                           x_test_tword,

                                           test_y,

                                           cv = 10).mean()

print("Word-Level TF-IDF Accuracy Score:", accuracy)
```

– **Tf-Idf N-gram Level**

```
xgb = xgboost.XGBClassifier()

xgboost_model = xgb.fit(x_train_tngram,train_y)

accuracy = model_selection.cross_val_score(xgboost_model,

                                           x_test_tngram,

                                           test_y,

                                           cv = 10).mean()

print("N-GRAM TF-IDF Accuracy Score:", accuracy)
```

– **Tf-Idf Char Level**

```
xgb = xgboost.XGBClassifier()

xgboost_model = xgb.fit(x_train_tchars,train_y)

accuracy = model_selection.cross_val_score(xgboost_model,

                                           x_test_tchars,

                                           test_y,

                                           cv = 10).mean()

print("CHARLEVEL Accuracy Score:", accuracy)
```

## 7.5 COMPARISON OF MACHINE LEARNING ALGORITHM ACCURACY

TWEETS 2017

Accuracy Score of Tweets 2017 shown in Table 7.13

**Table 7.13. Accuracy Score of Tweets 2017**

Data	Count Vector	Word Level	Ngram(1,2)	Ngram(2,3)	Char (1,5)	Char (2,10)
<b>SVM</b>	0.81709183 67346939	0.804591836 7346939	0.7921343537 414967	0.7630102040 816327	0.7838010204 081634	0.783843537 4149659
<b>Log.Reg.</b>	0.81505102 04081632	0.746386054 4217688	0.7401360544 217687	0.7380527210 884353	0.7318027210 884354	0.733886054 4217687
<b>N.Bayes</b>	0.79834183 67346939	0.731802721 0884354	0.7318027210 884354	0.7359693877 55102	0.7318027210 884354	0.731802721 0884354
<b>R. Forests</b>	0.81292517 00680273	0.785926870 7482993	0.7776360544 217688	0.7713010204 081633	0.8046343537 414966	0.792176870 7482993
<b>XgBoost</b>	0.79621598 63945578	0.800382653 0612244	0.7878826530 612244	0.7671343537 414966	0.7671768707 482993	0.783886054 4217688

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: SVM > LR > RF > NB > XGBoost

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(2,3): RF > XGBoost > SVM > LR > NB

Tf-idf Char(1,5): RF > SVM > XGBoost > LR = NB

Tf-idf Char(2,10): RF > XGBoost > SVM > LR > NB

TWEETS 2018

Accuracy Score of Tweets 2018 shown in Table 7.14

**Table 7.14. Accuracy Score of Tweets 2018**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Ngram(2,3)</b>	<b>Char(1,5)</b>	<b>Char(2,10)</b>
<b>SVM</b>	0.82028560 25039124	0.810622065 7276996	0.8050469483 568075	0.7716158059 467918	0.7966353677 621283	0.796674491 3928012
<b>Log.Reg.</b>	0.82034428 79499218	0.759076682 3161189	0.7465375586 854461	0.7395735524 256652	0.7521322378 716745	0.743740219 0923319
<b>N.Bayes</b>	0.80219092 33176838	0.743740219 0923319	0.7437402190 923319	0.7409624413 145541	0.7437402190 923319	0.743740219 0923319
<b>R.Forests</b>	0.79115805 94679186	0.796733176 8388106	0.7688184663 536777	0.7576877934 2723	0.7981416275 43036	0.777210485 1330204
<b>XGgBoost</b>	0.80637715 17996872	0.807746478 8732394	0.7938380281 69014	0.7549295774 647888	0.8065336463 223787	0.788419405 3208138

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: LR > SVM > XGBoost > NB > RF

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(2,3): SVM > RF > XGBoost > NB > LR

Tf-idf Char(1,5): XGBoost > RF > SVM > LR > NB

Tf-idf Char(2,10): SVM > XGBoost > RF > LR = NB



TWEETS 2019

Accuracy Score of Tweets 2019 shown in Table 7.15

**Table 7.15. Accuracy Score of Tweets 2019**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Ngram(2,3)</b>	<b>Char(1,5)</b>	<b>Char(2,10)</b>
<b>SVM</b>	0.83100100 99876557	0.827993491 1906631	0.8016777017 16979	0.7227864437 212433	0.8009089888 901357	0.773106273 1455504
<b>Log.Reg.</b>	0.82422848 16518909	0.761093031 0851756	0.7370497138 368308	0.7054819885 534732	0.7430535293 457524	0.715267646 7287622
<b>N.Bayes</b>	0.78513073 72909886	0.705498821 6810683	0.6904612276 961059	0.6912074963 528224	0.6851980698 013691	0.685198069 8013691
<b>R. Forests</b>	0.80545393 33408147	0.796425765 9073055	0.7611042531 702389	0.7039894512 400403	0.7663505779 373808	0.760357984 5135227
<b>XgBoost</b>	0.82650095 38772304	0.818219055 1004377	0.8159802491 302883	0.7190382673 100661	0.7798563573 111884	0.785113904 1633936

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: SVM > XGBoost > LR > RF > NB

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1,2): XGBoost > SVM > RF > LR > NB

Tf-idf N-gram(2,3): SVM > XGBoost > RF > LR > NB

Tf-idf Char(1,5): RF > SVM > XGBoost > LR = NB

Tf-idf Char(2,10): XGBoost > SVM > RF > LR > NB

TWEETS 2020

Accuracy Score of Tweets 2020 shown in Table 7.16

**Table 7.16. Accuracy Score of Tweets 2020**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Ngram(2,3)</b>	<b>Char(1,5)</b>	<b>Char(2,10)</b>
<b>SVM</b>	0.82728585 51704705	0.820181741 3355876	0.8124542124 542126	0.7388665821 358129	0.7792441532 826148	0.772693012 1160891
<b>Log.Reg.</b>	0.82793040 29304029	0.750753733 446041	0.7305684699 915469	0.7068223443 223444	0.7305579036 348266	0.713366441 2510566
<b>N. Bayes</b>	0.80177514 7928994	0.705043674 2744435	0.6979254719 639336	0.7020674837 98253	0.6961432797 971259	0.697921949 8450269
<b>R.Forests</b>	0.80000352 21189068	0.788739785 8551705	0.7626021414 482953	0.7210587489 433643	0.7721329952 099184	0.759058889 8281206
<b>XgBoost</b>	0.81900183 15018316	0.819001831 5018316	0.8112813468 58269	0.7186777965 624119	0.7620456466 610313	0.767371090 4480135

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: LR > SVM > XGBoost > NB > RF

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(2,3): SVM > RF > XGBoost > LR > NB

Tf-idf Char(1,5): SVM > RF > XGBoost > LR > NB

Tf-idf Char(2,10): SVM > XGBoost > RF > LR > NB

TWEETS 2017-2020

Accuracy Score of Tweets 2017-2020 shown in Table 7.17

**Table 7.17. Accuracy Score of Tweets 2017-2020**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Ngram(2,3)</b>	<b>Char(1,5)</b>	<b>Char(2,10)</b>
<b>SVM</b>	0.88321925 90424514	0.874209453 9068568	0.8601974535 916515	0.7441287388 411704	0.8433435399 804123	0.831476624 1514786
<b>Log.Reg.</b>	0.88393297 38492193	0.825777600 1621056	0.7878043700 960251	0.7237113169 951932	0.7984780088 03233	0.762397136 1349079
<b>N. Bayes</b>	0.81651056 50054597	0.718256014 2292668	0.7002150150 285373	0.7106612556 427374	0.6957047652 283549	0.696179824 6107778
<b>R.Forests</b>	0.85260382 07382557	0.842159268 7237563	0.8082105346 106652	0.7429360245 860116	0.8041787213 923067	0.791126971 4401504
<b>XgBoost</b>	0.87349123 61675542	0.860429917 4837614	0.8639911742 522317	0.7384314034 514977	0.8274515653 319223	0.838565498 749545

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: LR > SVM > XGBoost > NB > RF

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(2,3): SVM > RF > XGBoost > LR > NB

Tf-idf Char(1,5): SVM > RF > XGBoost > LR > NB

Tf-idf Char(2,10): SVM > XGBoost > RF > LR > NB

ALL TWEETS

Accuracy Score of All Tweets shown in Table 7.18

**Table 7.18. Accuracy Score of All Tweets**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Ngram(2,3)</b>	<b>Char(1,5)</b>	<b>Char(2,10)</b>
<b>SVM</b>	0.92379187 66346314	0.912287637 1753572	0.9029978078 676166	0.7764417057 641497	0.8838942875 346977	0.868905582 3631365
<b>Log.Reg.</b>	0.92073157 96274267	0.875661426 9356039	0.8432564082 000009	0.7525858919 350024	0.8450489888 121048	0.808844385 3340997
<b>N.Bayes</b>	0.85286068 82048129	0.756808197 3275589	0.7435083697 574841	0.7432973992 933493	0.7411865807 635928	0.741397662 6165685
<b>R.Forests</b>	0.89993650 83608464	0.894553085 6936628	0.8687997629 645471	0.8699613257 945366	0.8295347510 904966	0.831960465 872687
<b>XgBoost</b>	0.90521522 55178466	0.899304488 0791663	0.8993047108 568474	0.7663065465 449408	0.8794600091 784405	

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: SVM > LR > XGBoost > RF > NB

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1,2): SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(2,3): RF > SVM > XGBoost > LR > NB

Tf-idf Char(1,5): SVM > XGBoost > LR > RF > NB

Tf-idf Char(2,10): SVM > RF > LR > NB

Test Size = 0.25 (Default Size)

Accuracy Score of All Tweets shown in Table 7.19

**Table 7.19. Accuracy Score of All Tweets According to %25 Test Size**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Char(1,5)</b>
<b>SVM</b>	0.9237918766346314	0.9122876371753572	0.9029978078676166	0.8838942875346977
<b>Log. Reg.</b>	0.9207315796274267	0.8756614269356039	0.8432564082000009	0.8450489888121048
<b>N.Bayes</b>	0.8528606882048129	0.7568081973275589	0.7435083697574841	0.7411865807635928
<b>R.Forests</b>	0.8999365083608464	0.8945530856936628	0.8687997629645471	0.8295347510904966
<b>XgBoost</b>	0.9052152255178466	0.8993044880791663	0.8993047108568474	0.8794600091784405

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: SVM > LR > XGBoost > RF > NB

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf Char(1,5): SVM > XGBoost > LR > RF > NB

Test Size = 0.33

Accuracy Score of All Tweets shown in Table 7.20

**Table 7.20. Accuracy Score of All Tweets According to %33 Test Size**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Char(1,5)</b>
<b>SVM</b>	0.9318728057553957	0.9223579216626698	0.9131623980815349	0.8896529496402877
<b>Log.Reg.</b>	0.9297146602717825	0.8884547402078337	0.8635061231015188	0.8569488729016786
<b>N.Bayes</b>	0.8589464428457235	0.7672319744204638	0.7458817585931256	0.7429232933653077
<b>R.Forests</b>	0.907964892086331	0.904366554756195	0.876216274980016	0.8387178896882495
<b>XgBoost</b>	0.9057258513189449	0.9061258513189447	0.9040470663469226	0.8879737809752198

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: SVM > LR > RF > XGBoost > NB

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf Char(1,5): SVM > XGBoost > LR > RF > NB

Test Size = 0.40

Accuracy Score of All Tweets shown in Table 7.21

**Table 7.21. Accuracy Score of All Tweets According to %40 Test Size**

<b>Data</b>	<b>Count Vector</b>	<b>Word Level</b>	<b>Ngram(1,2)</b>	<b>Char(1,5)</b>
<b>SVM</b>	0.9379906301975846	0.9311301671064204	0.9238078319705322	0.9002565375271037
<b>Log.Reg.</b>	0.9381890418593312	0.89847605736827	0.8761791060372529	0.8646345689977968
<b>N.Bayes</b>	0.8616661006470039	0.7750508546896906	0.7454979666832118	0.7420674521277985
<b>R.Forests</b>	0.9140439492498063	0.9090969809381994	0.8818509713768211	0.8439213406828809
<b>XgBoost</b>	0.9110763081585203	0.9104820745926835	0.9092291247594415	0.8965625625887128

Comparison of Machine Learning Algorithm According to Each Feature Extraction

Count Vector: LR > SVM > RF > XGBoost > NB

Tf-idf Word Level: SVM > XGBoost > RF > LR > NB

Tf-idf N-gram(1.2): SVM > XGBoost > RF > LR > NB

Tf-idf Char(1,5): SVM > XGBoost > LR > RF > NB

## 8. CONCLUSION

In this thesis, the sentiment analysis of the tweets of Donald Trump, the 45th President of the United States of America, and the measurement and comparison of the success of the classification prediction according the feature extraction with machine learning algorithm have been made. The data set divided according to years from 2017 to 2020, representing Donald Trump's Presidency. In addition, sentiment analysis was performed on the Donald Trump's Presidency term (2017-2020) and whole tweet data set from 2009-05-04 to 2020-11-06.

TextBlob lexicon used for sentiment analysis and the sentiment polarity calculated for each tweet text and assigned to sentiment\_score. According to sentiment\_score each tweet labeled as Negative, Neutral and Positive. The Negative and Positive labeled sentiment are used for Binary Classification. Count Vector, Tf-Idf N-gram level, Tf-idf Char level and Tf-Idf Word level used as a feature extraction.

The estimation success of sentiment(sense\_class) classifications made with each feature extraction was compared with machine learning algorithms such as Support Vector Machine, Logistic Regression, Naive Bayes, Random Forest and XGBoost.

Considering the years and all the data, it seems that the Count Vector has the highest accuracy. Accuracy success ranking in feature extraction methods is from large to small, such as Count Vector, Tf-idf Word level, Tf-idf N-gram level and Tf-idf Char level.

When examine the tweet data by years, it has been observed that each machine learning algorithm shows a different success order in each year. In this way, the different accuracy success of each algorithm suggests that it is related to the structure of the data set and the number of samples it contains.

When compare the accuracy rates of the whole tweet data set with different test sizes (0.25, 0.33, 0.40) with machine learning algorithms, it is observed that the highest



accuracy machine learning algorithm is Support Vector Machine and the lowest accuracy machine learning algorithm is Naive Bayes. As can be seen from Pang's study, the Support Vector Machine algorithm has been the best performing algorithm. According to the observations made in the study, it was observed that the accuracy scores of machine learning algorithms increased as the test size increased.



## REFERENCES

- [1] Pang, B., Lee, L., and Vaithyanathan, S., 2002. *Thumbs up?: Sentiment Classification Using Machine Learning Techniques*, In Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10, July, USA, pp. 79-86.
- [2] Davidov D., Tsur O., Rappaport A. 2010. *Enhanced Sentiment Learning Using Twitter Hashtags and Smileys*, ICNC / Institute of Computer Science The Hebrew University. Coling 2010: Poster Volume, pp 241–249, Beijing, August
- [3] Çetin, M., Amasyalı, M.F., 2013 *Supervised and Traditional Term Weighting Methods for Sentiment Analysis*, Signal Processing and Communications Applications Conference (SIU), Haspolat, KKTC, pp 1-4
- [4] Meral, M., Diri, B., 2014. *Sentiment Analysis on Twitter*, IEEE 22nd Signal Processing and Communications Applications Conference (SIU), Trabzon, Türkiye, pp. 690-693.
- [5] Şafak H.İ. 2017. *Makine Öğrenmesi Nedir? (What is Machine Learning?)* | Medium.com [Online] Available at: <<https://medium.com/t%C3%BCrkiye/makine-%C3%B6C4%9Frenmesi-nedir-20dee450b56e>> [Accessed at 12 December 2020]
- [6] Murphy, K. P. Machine Learning A Probabilistic Perspective, London: The MIT Press.
- [7] Thomas W. Edgar, David O. Manz, in Research Methods for Cyber Security, 2017. *Machine Learning* | Science Direct [Online] Available at: <<https://www.sciencedirect.com/topics/computer-science/machine-learning#:~:text=Machine%20learning%20is%20a%20field,statistics%20and%20artificial%20intelligences%20communities>> [Accessed at 12 December 2020]
- [8] Potentia Analytics.com 2019. *What is Machine Learning: Definition, Types, Applications And Examples* | Potentia Analytics. [Online] Available at: <<https://www.potentiaco.com/what-is-machine-learning-definition-types-applications-and-examples/>>
- [9] Kumar GN C. 2018. *Machine Learning Types and Algorithms* | Towards Data Science [Online] Available at: <<https://towardsdatascience.com/machine-learning-types-and-algorithms-d8b79545a6ec>> [Accessed at 12 December 2020]
- [10] Mayo M. 2018. Frameworks for Approaching the Machine Learning Process | KDnuggets [Online] Available at: <<https://www.kdnuggets.com/2018/05/general-approaches-machine-learning-process.html>> [Accessed at 12 December 2020]
- [11] Rijmenam M. 2019. *7 Steps to Machine Learning: How to Prepare for an Automated Future* | Medium.com [Online] Available at: <<https://medium.com/dataseries/7-steps-to-machine-learning-how-to-prepare-for-an-automated-future-78c7918cb35d>> [Accessed at 12 December 2020]

- [12] Mary T. 2020. *Text Mining : A Basic Beginner's Guide* | Budding Data Scientist [Online] Available at: <<https://medium.com/budding-data-scientist/text-mining-a-basic-beginners-guide-5d2ecc04e4e6#:~:text=According%20to%20Wikipedia%2C%20Text%20Mining,patterns%20from%20text%2Dbased%20data>>
- [13] Ahlgren M. 2020. *Twitter İstatistikler ve 2020 İçin 50 + Gerçekler (Twitter Statistics and 50+ Facts For 2020)* | Websitehostingrating.com [Online] Available at: <<https://www.websitehostingrating.com/tr/twitter-statistics/>> [Accessed at 21 December 2020]
- [14] Subramanian N. B. 2020. *Converting Raw Text to Numerical Vectors using Bag of Words, N\_Grams and TF-IDF* | Aspirant.com [Online] Available at: <<https://aiaspirant.com/bag-of-words/>> [Accessed at 22 December 2020]
- [15] Sung M. 2019. *Character ngram tf-idf vectorizer using scikit learn* | mjeensung.github.io [Online] Available at: <<https://mjeensung.github.io/characterbigramtfidf/>> [Accessed at 22 December 2020]
- [16] Seker S.E. 2016. *Duygu Analizi (Sentimental Analysis)* | YBS Ansiklopedi, cilt 3, no. 3, pp. 21-36 [Online] Available at: <[http://ybsansiklopedi.com/wp-content/uploads/2016/09/duygu\\_analizi.pdf](http://ybsansiklopedi.com/wp-content/uploads/2016/09/duygu_analizi.pdf)>
- [17] Derici E. 2020. *Duygu Analizi Nedir, Kullanım Alanları ve Zorlukları (What is Sentiment Analysis, Its Uses and Difficulties)* | artwise.com [Online] Available at: <<https://www.artiwise.com/2020/07/13/sentiment-analysis/>> [Accessed at 26 December 2020]
- [18] Medhat W., Hassan A., Korashy H. 2014 *Sentiment analysis algorithms and applications: A survey* | Ain Shams Engineering Journal Volume 5, Issue 4, pp 1093-1113 ScienceDirect.com [Online] Available at: <<https://www.sciencedirect.com/science/article/pii/S2090447914000550>> [Accessed at 26 December 2020]
- [19] MonkeyLearn 2020. *Sentiment Analysis: A Definitive Guide* | MonkeyLearn.com [Online] Available at: <[https://monkeylearn.com/sentiment-analysis/#:~:text=Sentiment%20analysis%20\(or%20opinion%20mining,feedback%2C%20and%20understand%20customer%20needs,](https://monkeylearn.com/sentiment-analysis/#:~:text=Sentiment%20analysis%20(or%20opinion%20mining,feedback%2C%20and%20understand%20customer%20needs,)> [Accessed at 26 December 2020]
- [20] Boldenthusiast. 2019. *Sentiment Analysis – The Lexicon Based Approach* | Alphabold.com [Online] Available at: <<https://alphabold.com/sentiment-analysis-the-lexicon-based-approach/>> [Accessed at 26 December 2020]
- [21] Shah P. 2020. *Sentiment Analysis using TextBlob* | Towards Data Science [Online] Available at: <<https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob-3ac3a11d524>> [Accessed at 26 December 2020]

- [22] Ozel H. 2019. *Python Programlama Dili* | Dataficial TR [Online] Available at: <<https://medium.com/datamming/python-programlama-dili-1f1d88ef1e1d>> [Accessed at 12 December 2020]
- [23] Kaplan K. 2019. *Python ve Anaconda Kurulumu* | Kodcular [Online] Available at: <<https://medium.com/kodcular/python-ve-anaconda-kurulumu-b8931bd80e64>> [Accessed at 12 December 2020]
- [24] Yesilyurt A. 2018. *Veri Bilimi için Python Kütüphaneleri* | Amine Yesilyurt [Online] Available at: <<https://medium.com/@amine.yesilyurt/python-k%C3%BCt%C3%BCphaneleri-e59fe08cc276>> [Accessed at 12 December 2020]
- [25] Rao A. 2020. *Top 10 Python Libraries You Must Know In 2020* | Edureka.co [Online] Available at: <<https://www.edureka.co/blog/python-libraries/#z10>> [Accessed at 12 December 2020]
- [26] Yıldırım K. 2020. *Jupyter Lab Nedir? (What is Jupyter Lab?)* | Mekatronik Platformu [Online] Available at: <<https://www.mekatronikmuhendisligi.com/jupyter-lab-nedir.html>> [Accessed at 12 December 2020]
- [27] Hatipoglu E. 2018. *Machine Learning — Classification — Support Vector Machine— Kernel Trick— Part 10* | Ekrem Hatipoglu [Online] Available at: <<https://medium.com/@ekrem.hatipoglu/machine-learning-classification-support-vector-machine-kernel-trick-part-10-7ab928333158>> [Accessed at 27 December 2020]
- [28] Swaminathan S. 2018. *Logistic Regression — Detailed Overview* | Towards Data Science [Online] Available at: <<https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc>> [Accessed at 29 December 2020]
- [29] DevHunteryz.wordpress.com 2019. *Naive-Bayes Sınıflandırıcı (Naive Bayes Classifier)* | DevHunter Yapay Zeka, Robotik ve Sinirbilim [Online] Available at: <<https://devhunteryz.wordpress.com/2019/12/02/naive-bayes-siniflandirici/>> [Accessed at 28 December 2020]
- [30] Hatipoglu E. 2018. *Machine Learning — Classification — Naive Bayes — Part 11* | Ekrem Hatipoglu [Online] Available at: <<https://medium.com/@ekrem.hatipoglu/machine-learning-classification-naive-bayes-part-11-4a10cd3452b4>> [Accessed at 28 December 2020]
- [31] Wikipedia.org 2020. *Random Forest* | Wikipedia [Online] Available at: <[https://en.wikipedia.org/wiki/Random\\_forest](https://en.wikipedia.org/wiki/Random_forest)> [Accessed at 30 December 2020]
- [32] Wikipedia.org 2020. *XGBoost* | Wikipedia [Online] Available at: <<https://en.wikipedia.org/wiki/XGBoost>> [Accessed at 30 December 2020]

## ANNEX A

### A.1 Examples of the Numeric System

#### EQUATIONS:

**Equations 4.1** Inverse Document Frequency, Term Frequency - Inverse Document Frequency Equation

**Equations 6.1** Logistic Regression Formula

**Equations 6.2** Probability Formula, The probability of event A occurring when event B occurs

**Equation 6.3** Naïve Bayes Formula

**Equation 6.4** Accuracy Score

## **CURRICULUM VITAE**

### **Personal Information**

Name and surname: Kemal Mahmut KAŞGARLI

### **Academic Background**

Kadir Has University Faculty of Engineering and Natural Sciences  
Computer Engineering Department

Foreign Languages: English

### **Work Experience**

Uludağ University July 2010 – August 2015

Istanbul University August 2015-