INTRODUCTION

1.1 PROBLEM DEFINITION

The main objective of this project is to monitor the public feedback of the leaders on social media through tweets in twitter using R programming language. Here we compared two leaders Narendra Modi and Donald Trump in their respective reigning perspectives. For this we used sentiment analysis, which is a computational study of people's opinions, attitudes and emotions towards an entity. The algorithms which we used for classification of polarity and emotions are SVM (Support Vector Machine) and Naïve Bayes Algorithm. The results are graphically represented using bar graphs.

1.2 MOTIVATION OF THE PROJECT

We have chosen to work with twitter since we feel it is a better approximation of public sentiment as opposed to conventional internet articles and web blogs. The reason is that the amount of relevant data is much larger for twitter, as compared to traditional blogging sites. Moreover the response on twitter is more prompt and also more general (since the number of users who tweet is substantially more than those who write web blogs on a daily basis).

2. LITERATURE SURVEY

2.1 DATA MINING

Data Mining is an art and science of discovering and exploiting new, useful and profitable relationships in data. Data Mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems.

Data mining tools can answer business questions that traditionally were too time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations. Data mining techniques can be implemented rapidly on existing software and hardware platforms to enhance the value of existing information resources. Data Mining is also known as "Knowledge Discovery in Databases" process or KDD

2.2 TWITTER

Twitter is online news and social networking site where people communicate in short messages called tweets. It is also a free micro blogging service that allows registered members to tweet about a particular entity. It works on multiple platforms and devices.

2.2.1 Why Twitter?

We have chosen to work with twitter since we feel it is a better approximation of public sentiment as opposed to conventional internet articles and web blogs. The reason is that the amount of relevant data is much larger for twitter, as compared to traditional blogging sites. Moreover the response on twitter is more prompt and also more general (since the number of users who tweet is substantially more than those who write web blogs on a daily basis).

2.3 SENTIMENT ANALYSIS

2.3.1 Definition

Sentiment Analysis (SA) or Opinion Mining (OM) is the computational study of people's opinions, attitudes and emotions toward an entity. The entity can represent individuals, events or topics some researches stated that OM and SA have slightly different notions. Opinion Mining extracts and analyzes people's opinion about an entity while Sentiment Analysis identifies the sentiment expressed in a text then analyzes it. Therefore, the target of SA is to find opinions, identify the sentiments they express, and then classify their polarity.

2.3.2 Types of Sentiment Analysis

In sentiment analysis there are three main classification levels in SA: document-level, sentence-level, and aspect-level SA. Document-level SA aims to classify an opinion document as expressing a positive or negative opinion or sentiment. It considers the whole document a basic information unit (talking about one topic). Sentence-level SA aims to classify sentiment expressed in each sentence. The first step is to identify whether the sentence is subjective or objective. If the sentence is subjective, Sentence-level SA will determine whether the sentence expresses positive or negative opinions.

2.3.3 Sentiment Classification techniques

This can be roughly divided into machine learning approach, lexicon based approach and hybrid approach. The Machine Learning Approach applies the famous ML algorithms and uses linguistic features. The Lexicon-based Approach relies on a sentiment lexicon, a collection of known and precompiled sentiment terms. It is divided into dictionary-based approach and corpus-based approach which use statistical or semantic methods to find sentiment polarity. The hybrid Approach combines both approaches and is very common with sentiment lexicons playing a key role in the majority of methods. The text classification methods using ML approach can be roughly divided into supervised and unsupervised learning methods. The supervised methods make use of a large number of labeled training documents. The unsupervised methods are used when it is difficult to find these labeled training documents. The lexicon-based approach depends on finding the opinion lexicon which is used to analyze the text. There are two methods in this approach. The dictionary-based approach which depends on finding opinion seed words, and then searches the dictionary of their synonyms and antonyms. The corpus-based approach begins with a seed list of opinion words, and then finds other opinion words in a large corpus to help in finding opinion words with context specific orientations. This could be done by using statistical or semantic methods. There is a brief explanation of both approaches' algorithms and related articles in the next subsections.

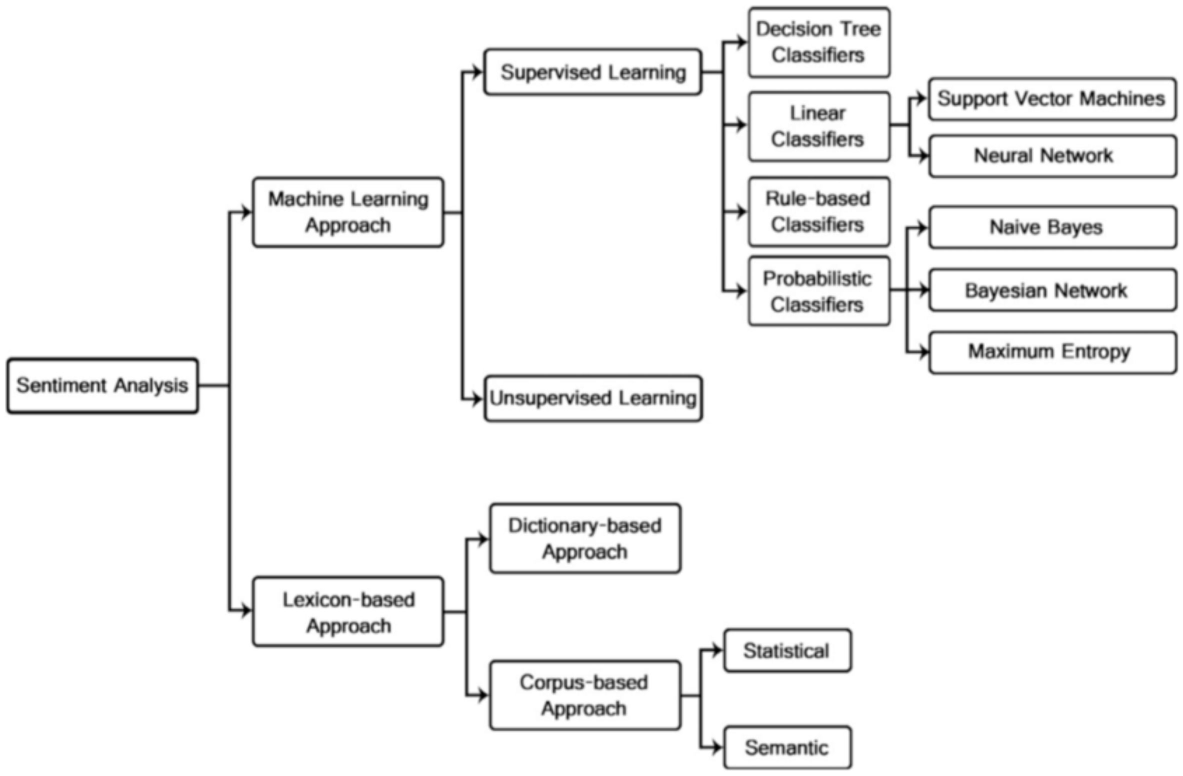


Fig:2 Sentiment Analysis Techniques

This Fig is about Sentiment Classification techniques, it can be roughly divided into machine learning approach, lexicon based approach and hybrid approach. The Machine Learning Approach is further divided into supervised and unsupervised learning. In our project we are using the Naïve Bayes algorithm and Support vector machines algorithm. The Lexicon-based Approach relies on a sentiment lexicon, a collection of known and precompiled sentiment terms.

2.4 Algorithms Used

2.4.1 Naïve Bayes

The Naive Bayes classifier is the simplest and most commonly used classifier. Naive Bayes classification model computes the posterior probability of a class, based on the distribution of the words in the document. The model works with the BOWs feature extraction which ignores the position of the word in the document. It uses Bayes Theorem to predict the probability that a given feature set belongs to a particular label.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:

$$\underbrace{P(c\mid x)}_{\text{Posterior Probability}} = \frac{\overbrace{P(x\mid c)}^{\text{Likelihood}}\,\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$$

$$P(c\mid \mathrm{X}) = P(x_1\mid c)\times P(x_2\mid c)\times\cdots\times P(x_n\mid c)\times P(c)$$

Fig:3 Navie Bayes formula

This Fig indicates the Navie Bayes formula to classify the probability

Above,

- *P*(*c*|*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.

- $P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.

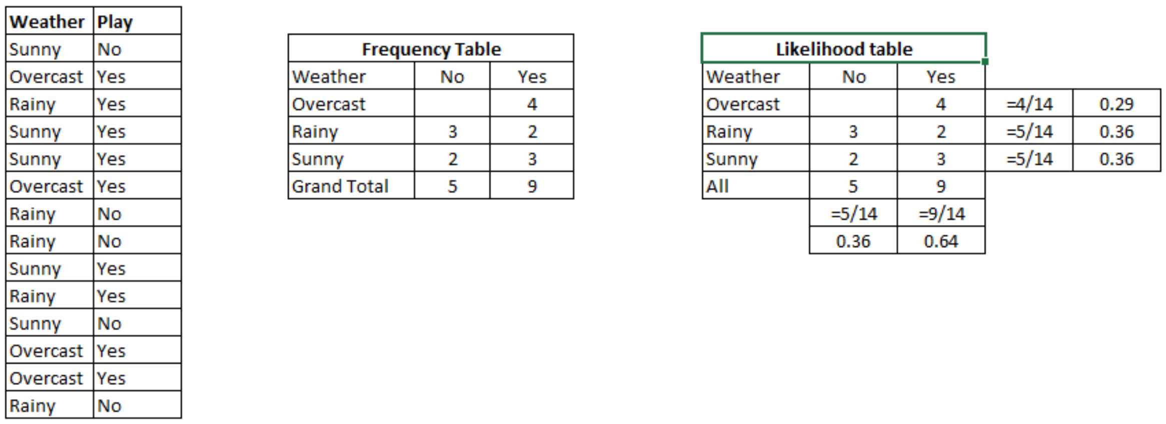
- $P(x)$ is the prior probability of *predictor*.

2.4.1.1 How Naive Bayes algorithm works?

Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play' (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set into a frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.



Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Fig:4 Example

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will play if weather is sunny. Is this statement is correct?

We can solve it using above discussed method of posterior probability.

P(Yes | Sunny) = P(Sunny | Yes) * P(Yes) / P (Sunny)

Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

2.1.4.2What are the Pros and Cons of Naive Bayes?

Pros:

- It is easy and fast to predict class of test data set. It also perform well in multi class prediction
- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
- It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons:

- If categorical variable has a category (in test data set), which was not observed in training data set, then model will assign a 0 (zero) probability and will be unable to make a prediction. This is often known as "Zero Frequency". To solve this, we can use the smoothing technique. One of the simplest smoothing techniques is called Laplace estimation.
- On the other side naive Bayes is also known as a bad estimator, so the probability outputs from predict_proba are not to be taken too seriously.

2.4.2 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well. Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

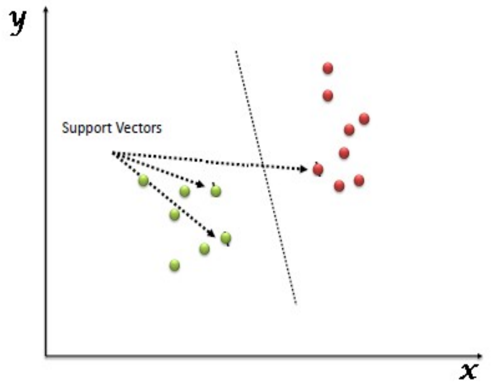


FIG:5 Svm

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression. mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well

2.4.2.1 What are the Pros and Cons of Support Vector Machine?

- **Pros:**
    - It works really well with clear margin of separation

- It is effective in high dimensional spaces.
- It is effective in cases where number of dimensions is greater than the number of samples.
- It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- **Cons:**
  - It doesn’t perform well, when we have large data set because the required training time is higher
  - It also doesn’t perform very well, when the data set has more noise i.e. target classes are overlapping
  - SVM doesn’t directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library.

3. ANALYSIS

3.1 INTRODUCTION

Requirement Analysis is the first phase in the software development process. The main objective of the phase is to identify the problem and the system to be developed. The later phases are strictly dependent on this phase and hence requirements for the system analyst to be clearer, precise about this phase. Any inconsistency in this phase will lead to lot of problems in the other phases to be followed. Hence there will be several reviews before the final copy of the analysis is made on the system to be developed. After all the analysis is completed the system analyst will submit the details of the system to be developed in the form of a document called requirement specification.

The Requirement analysis task is a process of discovery, refinement, modeling and specifications. The software scope, initially established by a system engineer and refined during software project planning, is refined in detail. Models of required data, information and control flow and operational behaviour are created. Alternative solutions are analyzed and allocated to various software elements.

Both the developer and the customer take an active role in requirement analysis and specification. The customer attempts to reformulate a sometimes-nebulous concept of software function and performance into concrete detail. The developer acts as interrogator, consultant and problem solver. The communication content is very high. Changes for misinterpretation of misinformation abound. Ambiguity is probable.

Requirement analysis is a software engineering task that bridges the gap between the system level software allocation and software design. Requirement analysis enables the system engineer to specify the software function and performance indicate software interface with other system elements and establish constraints that software must meet. It allows the software engineer, often called analyst in this role, to refine the software allocation and build model of the data, functional and behavioral domains and that will be treated by software. Requirements analysis provides the software designer with models that can be translated into data, architectural, interface and procedural design. Finally, the requirement specification provides the developer and customer with the means to access quality once software builds.

Software requirements analysis may be divided into five areas of effort

- Problem recognition
- Evaluation and Synthesis
- Modeling
- Review

Once the tasks in the above areas have been accomplished, the specification document would be developed which now forms the basis for the remaining software engineering tasks. Keeping this in view the specification document of the current system has been generated.

3.2 SOFTWARE REQUIREMENT SPECIFICATION

Requirements Determination is the process, by which analyst gains the knowledge of the organization and apply it in selecting the right technology for a particular application.

3.2.1 Introduction

A Software Requirements Specification (SRS) is a complete description of the behavior of the system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. Use cases are also known as functional requirements. In addition to use cases, the SRS also contains non-functional (or supplementary) requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

3.2.2 Functional Requirements

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases. Functional requirements are supported by non-functional requirements (also known as quality requirements), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). How a system implements functional requirements is detailed in the system design. In some cases a requirements analyst generates use cases after gathering and validating a set of functional requirements. Each use case illustrates behavioral scenarios through one or more functional requirements. Often, though, an analyst will begin by eliciting a set of use cases, from which the analyst can derive the functional requirements that must be implemented to allow a user to perform each use case.

3.2.3 Non Functional Requirements

In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. This should be contrasted with functional requirements that define specific behaviour or functions In general, functional requirements define what a system is supposed to do whereas non-functional requirements define how a system is supposed to be. Non-functional requirements are often called qualities of a system. Other terms for non-functional requirements are "constraints", "quality attributes", "quality goals" and "quality of service requirements," and "non-behavioral requirements." Qualities, that is, non-functional requirements, can be divided into two main categories:

1. Execution qualities, such as security and usability, which are observable at run time.

2. Evolution qualities, such as testability, maintainability, extensibility and scalability, which are embodied in the static structure of the software system.

3.3 SYSTEM REQUIREMENTS

3.3.1 Introduction

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These pre-requisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. With increasing demand for higher processing power and resources in newer versions of software, system requirements tend to increase over time. Industry analysts suggest that this trend plays a bigger part in driving upgrades to existing computer systems than technological advancements.

3.3.2 Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Minimum Hardware Requirements

1. VDU: Monitor / Projector
2. Input Devices: Keyboard and Mouse
3. RAM: 4 GB or more

4. Processor: P4 or above
5. Storage: 4GB or more

3.3.3 Software Requirements

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Minimum Software Requirements

1. Operating System: Windows 10
2. Programming language: R programming language(1.1.383)
3. Tools used: R studio(R 3.4.4)

4. DESIGN

4.1 Introduction:

Grady Booch, James Raumbaugh and Ivar Jacobson have collaborated to combine the best features of their individual object oriented analysis and design methods into a unified method the unified modeling language, the version 1.0 for the Unified Modeling was released in January 1997 the main parts of UML are based on the Booch, OMT and OOSE methods.

The goals of UML are:

- To model systems using object-oriented concepts
- To establish an explicit coupling between conceptual as well as executable
- To address the issues of scale inherent in complex, mission critical system
- To create a modeling language usable by both humans and machines

4.2 Basic Building Blocks Of UML

The basic building blocks in UML are things and relationships; these are combined in different ways following rules to create different types of diagrams. In UML these are nine types of diagrams, below is a list and brief description of them. The more in depth description in the document, will focus on the first five diagrams in the list, which can be seen as the most general, sometimes also referred to as the UML core diagrams**.**

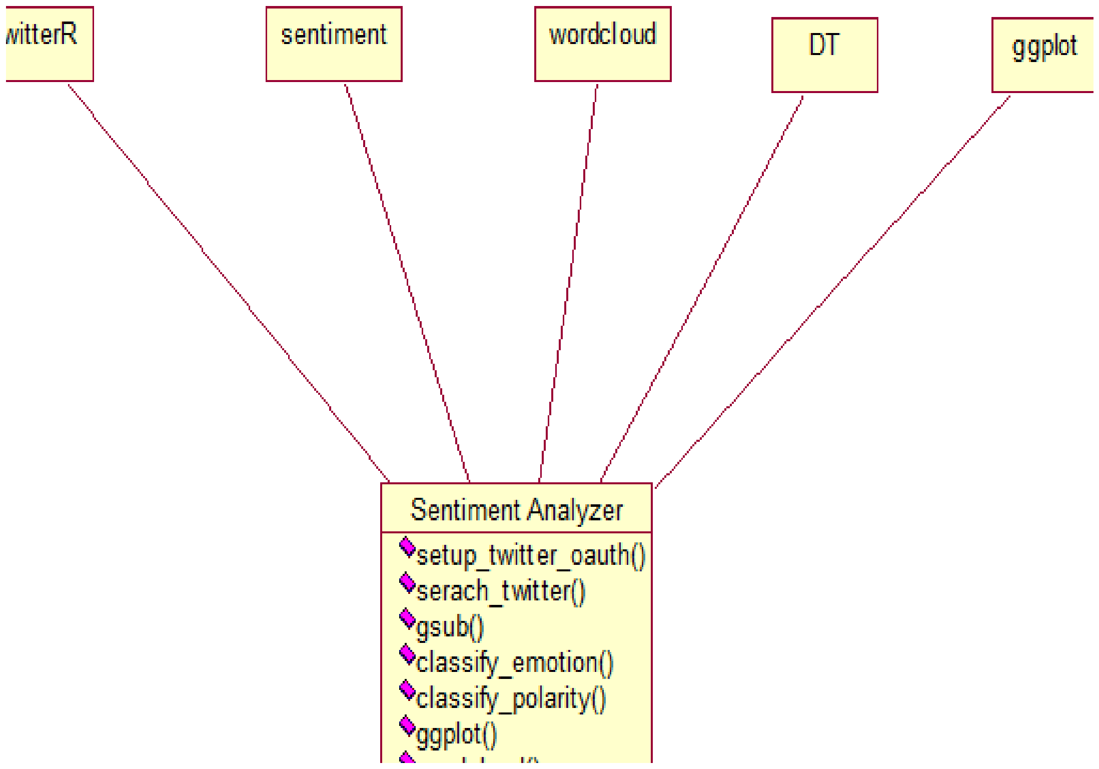
- **Use Case Diagrams**: it shows a set of use cases, and how actors can use them.
- **Class Diagrams**: describes the structure of the system, divided in classes with different connections and relationships.
- **Sequence Diagrams**: it shows the interaction between a set of object, through the messages that may be dispatched between them.

- **State Chart Diagrams**: state machines, consisting of states, transitions, events and activities.
- **Activity Diagrams**: shows the flow through a program from an defined start point to an end point.
- **Object Diagrams:** A set of objects and their relationships, this is a snapshot of instances of the things found in the class objects.
- **Collaboration Diagrams:** Collaboration diagrams emphasize structural ordering of objects that send and receive messages.
- **Component Diagrams:** shows organizations and dependencies among a set of components. These diagrams address the static implementation view of the system.
- **Deployment Diagrams:** show the configuration of run-time processing nodes and components that live on them.

4.2.1Class Diagram

The classes in the Class Diagram represent both the main objects, interactions in the applications and the classes to be programmed. In the diagram, classes are represented with the boxes which contain three parts.

- The top part contains the name of the class. It is printed in bold and centred, and the first letter is capitalized.
- The middle part contains the attributes of the class. They are left-aligned and the first letter is lowercase.
- The bottom part contains the methods the class can execute. They are also left-aligned and the first letter is lowercase.



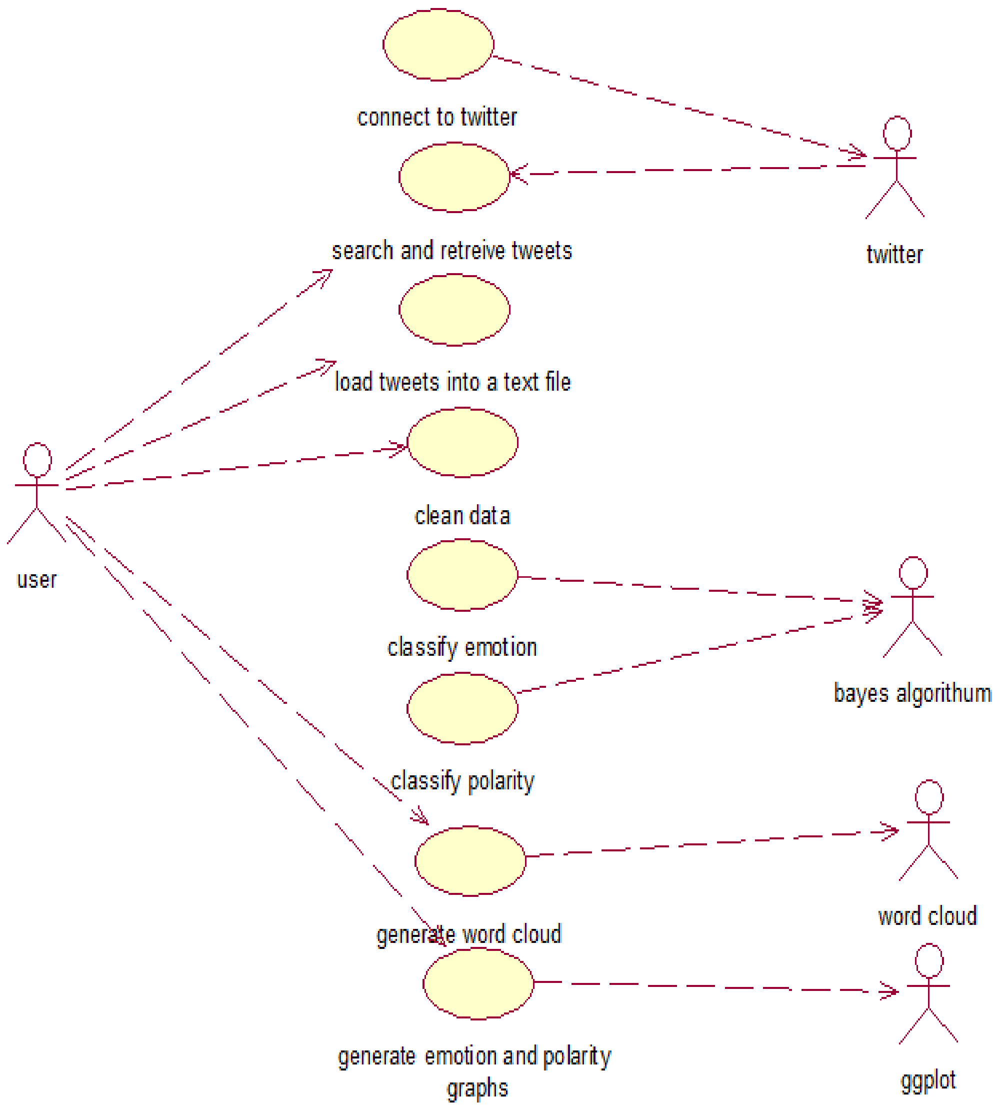
TwitterR, Sentiment, Word cloud, Data table and ggplot are the different classes used here. The sentiment analyzer contains certain functions. They are as follows:

- Setup_twitter_oauth() – this function is used to provide authentication to all the accounts in twitter.
- Search_twitter() – this function searches for the tweets in twitter.
- Gsub() – this function replaces all matches of a string.
- Classify_emotion and classify_polarity – these functions classifies the tweets into emotions and polarity.
- Ggplot() – is used to form graphs.

4.2.2 Use Case Diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. Uses cases are the system functionalities written in an organized manner. The things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system.

4.2.3 Sequence Diagram

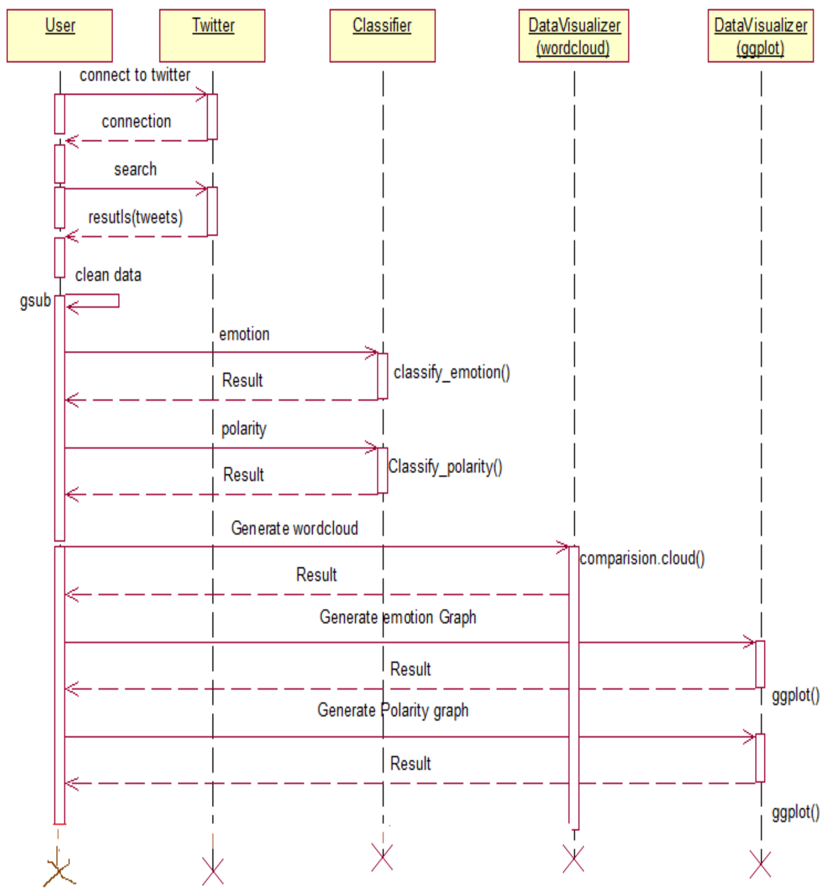


Firstly, the user connects to the twitter, then Twitter searches and retrieves for the tweets. Then the user loads tweets into text file and performs data cleaning. Bayes algorithm classifies the emotions and polarity. Word cloud is generated later. GGplot is a function which helps in generating graphs of polarity and emotions.

4.2.3 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

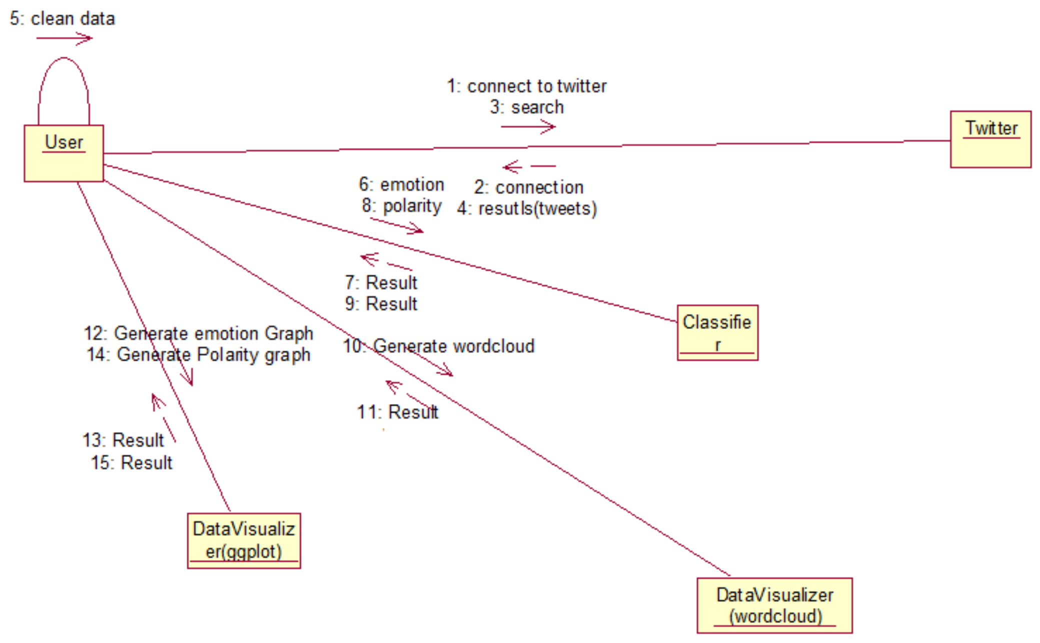
A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur.



In the above figure, the user connects to the twitter, the dotted line represents the response for the request. Gsub is a self loop which continues until the whole data is cleaned. Later emotions and polarity are classified. The word cloud is generated and hence the emotions and polarity graphs are formed as a result.

4.2.4 Collaboration Diagram

Collaboration diagram emphasize structural ordering of objects that send and receive messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



This diagram stresses on the structural ordering of objects that send and receive messages. Collaboration diagram takes information from other diagrams like class diagram, use case diagrams etc. The numbers are given to known which process is next.

4.2.5 Activity Diagram

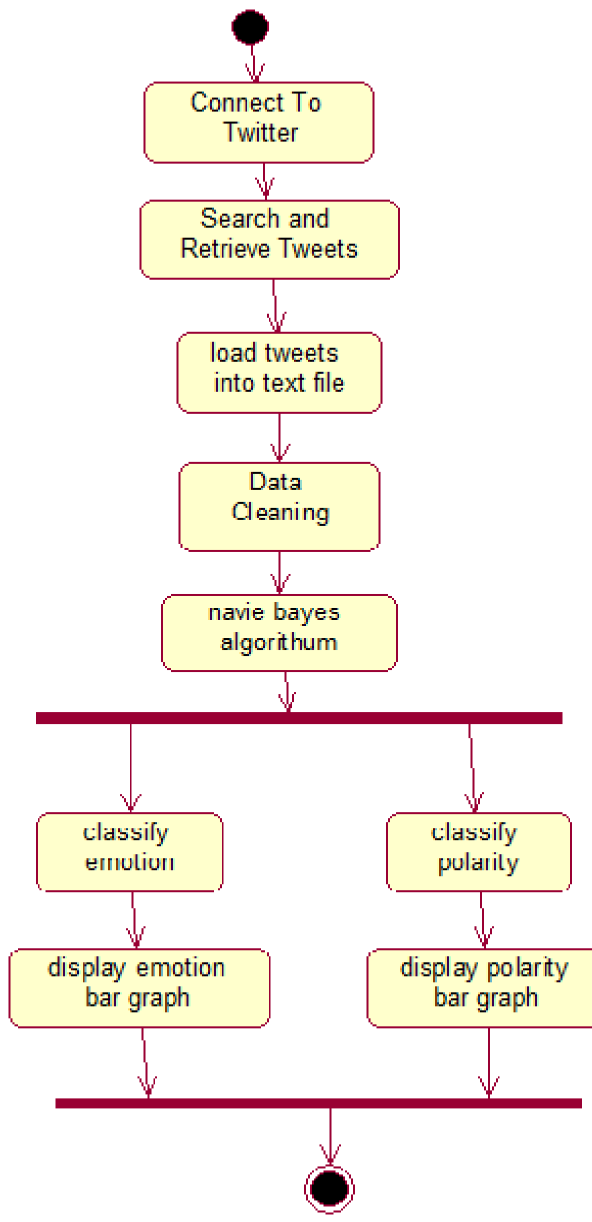

Connect To Twitter
Search and Retrieve Tweets
load tweets into text file
Data Cleaning
navie bayes algorithum
classify emotion
classify polarity
display emotion bar graph
display polarity bar graph

Fig:10 Activity Diagram

Activity diagram represents the flow of the project, the start point represents the start of the process. The links in middle represent the relationship between them. It is an interactive process and goes in a sequential manner one after the other.

4.2.6 Deploymemt Diagram

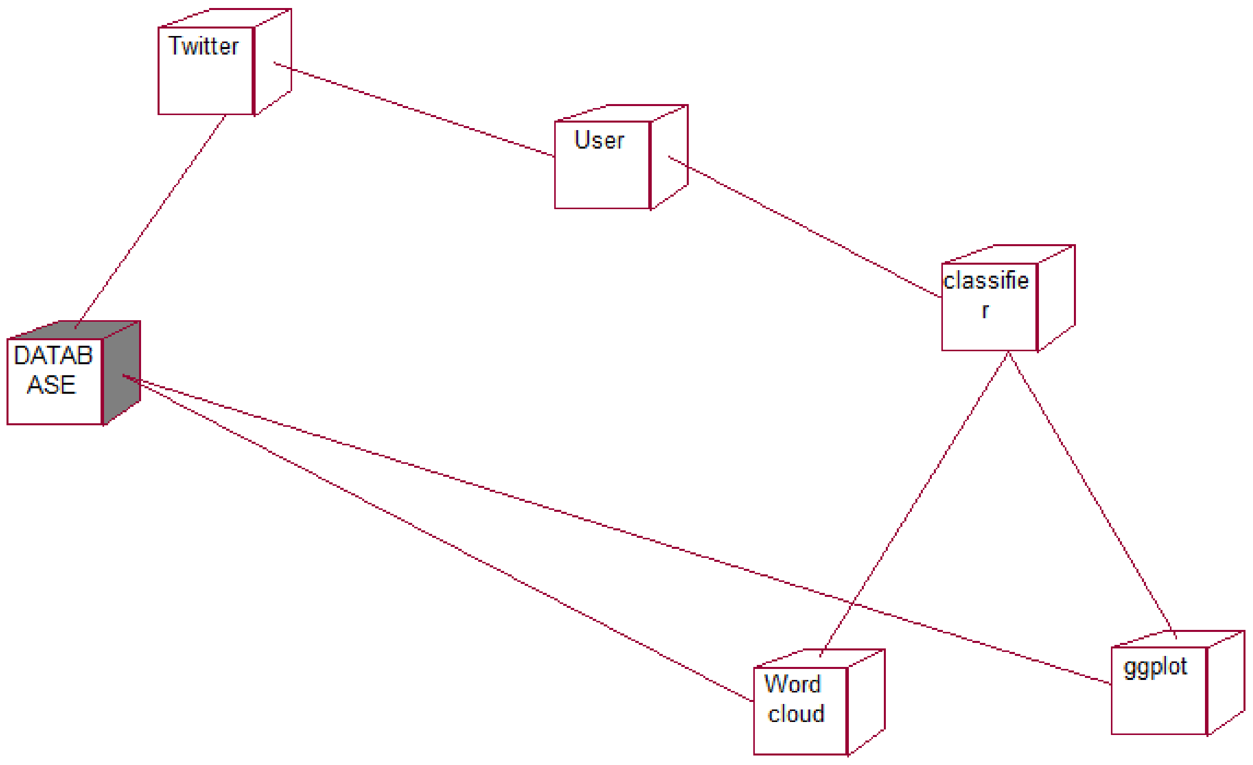


Fig:11 Deployment Diagram

This diagram represents the run time processing nodes and components that live on them. The word cloud and the plots are connected to the node database so as to store the data. The user connects to the twitter and classifier to form word cloud and ggplot.

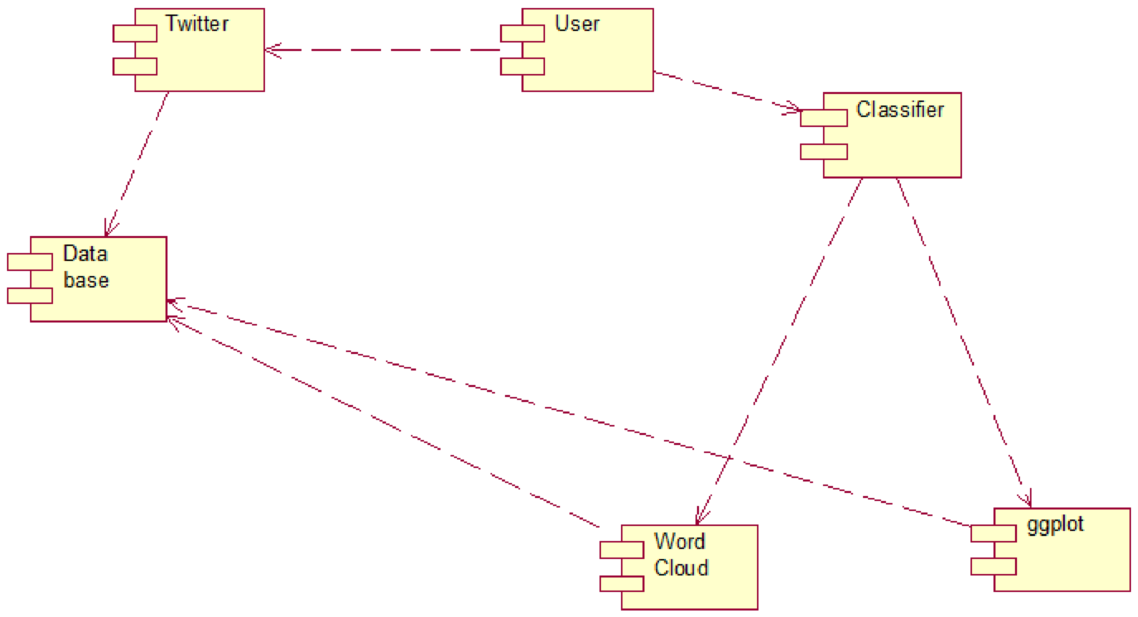


Fig:12 Component Diagram

5. IMPLEMENTATIONS AND RESULTS

5.1 Introduction

5.1.1 Flow of project

Step1: The first step of the project is to create a twitter account. After creating an account we have to generate the access keys.
Step2: For generating the access keys first we have to create an application in twitter. Four access keys will be generated. These keys will be useful in getting connected to Twitter and access the tweets.
Step3: After generating keys, we use library functions for using the functions of R and perform different types of actions.
Step4: Firstly, using these keys we extract 10,000 tweets for each domain ie., Narendra Modi and Donald Trump.
Step5: On this raw data set we have to perform data cleaning, a process in which we clean all the unnecessary and unwanted data.
Step6: Then we obtain the cleaned data set on which we perfom actions like classification of emotions and polarity using different algorithms like Naïve Bayes and SVM.
Step7: Emotion graph and polarity graphs will be generated in which there are categories like joy,sad,anger,digust,surprise and for polarity we have positive , negative and neutral.
Step8: After this,we generate the data tables in which we can view the tweets, emotions and the polarity. We can also search of the required tweet in the search box.
Step9: Word clouds can also be generated.The greater the size the more number of times the word has repeated.
Step10: In SVM we the same process will be followed. Scores graph will also be generated.

5.1.2 How to install R ?

1.Open an internet browser and go to http://cran.rproject.org.

2.Click the “download R” link in the middle of the page under “getting started.”

3.Select a CRAN location and click the corresponding link.

4.Click on the “download R for Windows” link at top of the page.

5.Click on “install R for the first time” link at the top of the page.

6.Cick “ Download R for Windows” and save the executable file somewhere on your computer. Run the .exe file and follow the installation instructions.

7.Now that R is installed you need to download and install Rstudio.

5.1.2 How to install RStudio

1. Go to www.rstudio.com and click on the "Download RStudio" button.

2. Click on "Download RStudio Desktop."

3. Click on the version recommended and run the .exe file and follow.

5.2 Software design

5.2.1 Front End Design:

HTML:

HTML is a format that tells a computer how to display a web page. The documents themselves are plain text files with special "tags" or codes that a web browser uses to interpret and display information on your computer screen. HTML stands for Hyper Text Markup Language; an HTML file is a text file containing small markup tags. The markup tags tell the Web browser how to display the page. An HTML files must have an htm or html file extension.

CSS:

Cascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of a web page. Using CSS, you can control the colour of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colours are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

PHP:

PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including

images, with the generated web page. PHP code may also be executed with a command-line interface(CLI) and can be used to implement standalone graphical applications.
The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge. The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a defacto standard. Since 2014 work has gone onto create a formal PHP specification.

5.3 Sample Codes

5.3.1 Navie Bayes Code:

```
library(devtools) #used as development tools

library(twitteR) #used for authenticating twitter tweeets

library(httr) #deals with http actions and links

library(sentiment) #used for performing sentiment analysis

library(DT) #used for generating data tables

library(wordcloud) #used for generating word clouds

if (!require("pacman")) install.packages("pacman")

pacman::p_load(devtools, installr)

api_key <- "t4f2zQp9044P92l5J6PCz9wW2"

api_secret <- "cEKCFf9YSRYMnqMnKXizZ4RKB8dixPdGXgdtYmLYrY1xYY4HAV"

access_token <- "2717578724-rzBS0rZcNBg6xecEpFukQTFP4DVd4adeOgrhGHT"

access_token_secret <- "MeUkekEJ2Hu5f8uD9YECWBMhjJ9YdQ3mC9H7jbcbd00l0"

setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)
```

#Step1: Tweets Extraction

```
modi_tweets <- searchTwitter("$narendramodi", n = 1000,lang='en')
```

```
namo_txt = sapply(modi_tweets, function(x) x$getText())

tweetsdf <- twListToDF(modi_tweets)

write.csv(tweetsdf, file='~/moditweets1k.csv',row.names = F)

trump_tweets <- searchTwitter("$realDonaldTrump", n = 1000,lang='en')

dtrump_txt = sapply(trump_tweets, function(x) x$getText())

tweetsdf <- twListToDF(modi_tweets)

write.csv(tweetsdf, file='~/moditweets1k.csv',row.names = F)
```

#Step2: Data Cleaning

```
namo_txt = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', namo_txt)

namo_txt = gsub('@\\w+', '', namo_txt)

namo_txt = gsub('[[:punct:]]', '', namo_txt)

namo_txt = gsub('[[:digit:]]', '', namo_txt)

namo_txt = gsub('http\\w+', '', namo_txt)

namo_txt = gsub('[ \t]{2,}', '', namo_txt)

namo_txt = gsub('^\\s+|\\s+$', '', namo_txt)

dtrump_txt = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', dtrump_txt)

dtrump_txt = gsub('@\\w+', '', dtrump_txt)

dtrump_txt = gsub('[[:punct:]]', '', dtrump_txt)

dtrump_txt = gsub('[[:digit:]]', '', dtrump_txt)

dtrump_txt = gsub('http\\w+', '', dtrump_txt)

dtrump_txt = gsub('[ \t]{2,}', '', dtrump_txt)

dtrump_txt = gsub('^\\s+|\\s+$', '', dtrump_txt)

try.error = function(x)

{

  y = NA
  try_error = tryCatch(tolower(x), error=function(e) e)
  if (!inherits(try_error, 'error'))
    y = tolower(x)
  return(y)
}
namo_txt = sapply(namo_txt, try.error)
namo_txt = namo_txt[!is.na(namo_txt)]
names(namo_txt) = NULL
dtrump_txt = sapply(dtrump_txt, try.error)
dtrump_txt = dtrump_txt[!is.na(dtrump_txt)]
names(dtrump_txt) = NULL
```

#Step3: Emotion and Polarity Graphs

```
class_emo = classify_emotion(namo_txt, algorithm='bayes', prior=1.0)
namo_emotion = class_emo[,7]
emotion[is.na(namo_emotion)] = 'unknown'
class_emo = classify_emotion(dtrump_txt, algorithm='bayes', prior=1.0)
dtrump_emotion = class_emo[,7]
emotion[is.na(dtrump_emotion)] = 'unknown'
class_pol = classify_polarity(namo_txt, algorithm='bayes')
namo_polarity = class_pol[,4]
namo_df= data.frame(text=namo_txt, namo_emotion=namo_emotion,
namo_polarity=namo_polarity, stringsAsFactors=FALSE)
class_pol = classify_polarity(dtrump_txt, algorithm='bayes')
dtrump_polarity = class_pol[,4]
dtrump_df = data.frame(text=dtrump_txt, dtrump_emotion=dtrump_emotion,
 dtrump_polarity=dtrump_polarity, stringsAsFactors=FALSE)

namo_df = within(namo_df, namo_emotion <- factor(namo_emotion,

 levels=names(sort(table(namo_emotion), decreasing=TRUE))))

namo_df = data.frame(text = namo_df, namo_emotion = namo_emotion,

namo_polarity = namo_polarity, stringsAsFactors = FALSE)

namo_df = within(namo_df,namo_emotion <- factor(namo_emotion, levels =

names(sort(table(namo_emotion), decreasing = TRUE))))

datatable(namo_df, class = "cell-border-stripe",  rownames = TRUE, colnames =
c(“No”,"tweet",

 "Emotion", "Polarity"), options = list(pageLength = 10))

dtrump_df = within(dtrump_df, dtrump_emotion <- factor(dtrump_emotion,

levels=names(sort(table(dtrump_emotion), decreasing=TRUE))))

dtrump_df = data.frame(text = dtrump_df, dtrump_emotion =
dtrump_emotion, dtrump_polarity =

 dtrump_polarity, stringsAsFactors = FALSE)

dtrump_df = within(dtrump_df, dtrump_emotion <- factor(dtrump_emotion, levels =

 names(sort(table(dtrump_emotion), decreasing = TRUE)))) datatable(dtrump_df, class =
"cell-border-

stripe", rownames = TRUE, colnames = c(“no”,"tweet", "Emotion", "Polarity"), options =
list(pageLength =

 10))

raw=c(namo_emotion,dtrump_emotion)

emoton_cat= c(namo_df,dtrump_df)

polaritytot= c(namo_polarity,dtrump_polarity)

namo_dataFrame= data.frame(raw,polaritytot, check.rows = FALSE)

levels(namo_emotion) = c(levels(namo_emotion),"None")

namo_emotion[is.na(namo_emotion)] = "None"
```

```
levels(dtrump_emotion) = c(levels(dtrump_emotion),"None")

dtrump_emotion[is.na(dtrump_emotion)] = "None"

library(stringr)

Emotions=
c(rep("anger",2),rep("disgust",2),rep("fear",2),rep("joy",2),rep("sadness",2),rep("surprise",2)
)

Candidates= rep(c("Donald Trump","Narendra Modi"),6)

Values=c(sum(str_count(namo_emotion,"anger")),sum(str_count(dtrump_emotion,"anger")),
sum(str_count

(namo_emotion,"disgust")),

        sum(str_count(dtrump_emotion,"disgust")), sum(str_count(namo_emotion,"fear")),

        sum(str_count(dtrump_emotion,"fear")),

        sum(str_count(namo_emotion,"joy")),

        sum(str_count(dtrump_emotion,"joy")),

        sum(str_count(namo_emotion,"sadness")),

        sum(str_count(dtrump_emotion,"sadness")),

        sum(str_count(namo_emotion,"surprise")),

        sum(str_count(dtrump_emotion,"surprise"))

        )

ggplot(data, aes(fill=Candidates, y=Values, x=Emotions)) +  geom_bar(position="dodge",
stat="identity")

Polarities= c (rep("positive",2), rep("negative",2), rep("nuetral",2))

Polarity_Candidates= rep(c("Narendra Modi","Donald Trump"),3)

Polarity_Values = c(sum(str_count(namo_polarity,"positive")),
sum(str_count(dtrump_polarity,"positive")),

               sum(str_count(namo_polarity,"negative")),
sum(str_count(dtrump_polarity,"negative")),
                    sum(str_count(namo_polarity,"neutral")),
sum(str_count(dtrump_polarity,"neutral")))

polarity_data= data.frame(Polarities,Polarity_Candidates,Polarity_Values)

ggplot(polarity_data, aes(fill=Polarity_Candidates, y=Polarity_Values, x=Polarities))

+ geom_bar(position="dodge", stat="identity")

plot1= ggplot(namo_df, aes(x=namo_emotion)) +

  geom_bar(aes(y=..count.., fill=namo_emotion)) +

  scale_fill_brewer(palette='Dark2') +

  labs(x='emotion categories', y='number of tweets') +

  ggtitle('Sentiment Analysis of Modi') +theme_minimal()


plot2= ggplot(dtrump_df, aes(x=dtrump_emotion)) +

  geom_bar(aes(y=..count.., fill=dtrump_emotion)) +

  scale_fill_brewer(palette='Dark2') +

  labs(x='emotion categories', y='number of tweets') +

  ggtitle('Sentiment Analysis of Trump') +theme_minimal()

grid.newpage()

grid.draw(rbind(ggplotGrob(plot1), ggplotGrob(plot2), size = "last"))
```

#ggplot for polarity

```
ggplot(namo_df, aes(x=polarity)) +

  geom_bar(aes(y=..count.., fill=polarity)) +

  scale_fill_brewer(palette='RdGy') +

  labs(x='polarity categories', y='number of tweets') +

  ggtitle('Sentiment Analysis of Tweets about Narendra Modi\n(classification by polarity)') +

  theme(plot.title = element_text(size=12, face='bold'))
```

```
emos = levels(factor(namo_df$emotion))

nemo = length(emos)

emo.docs = rep('', nemo)

for (i in 1:nemo)

{

  tmp = some_txt[emotion == emos[i]]

  emo.docs[i] = paste(tmp, collapse=' ')

}

emo.docs = removeWords(emo.docs, stopwords('english'))

emo.docs = removeWords(emo.docs, stopwords('german'))

emo.docs = removeWords(emo.docs, stopwords('french'))

corpus = Corpus(VectorSource(emo.docs))

tdm = TermDocumentMatrix(corpus)

tdm = as.matrix(tdm)

colnames(tdm) = emos
```

#Step4 : Word Cloud Generation

```
comparison.cloud(tdm, colors = brewer.pal(nemo, 'Dark2'),scale = c(3,.5), random.order =
FALSE,

title.size = 1.5)

emos = levels(factor(sent_df$emotion))

nemo = length(emos)

emo.docs = rep("", nemo)

for (i in 1:nemo)

{
  tmp = some_txt[emotion == emos[i]]

  emo.docs[i] = paste(tmp, collapse=" ")

}

comparison.cloud(tdm, colors = brewer.pal(nemo, "Dark2"), scale = c(3,.5), random.order =
FALSE,

title.size = 1.5)
```

5.3.2 SVM(SUPPORT VECTOR MACHINE) CODE:

5.3.2.1 NARENDRA MODI

```
library(devtools)
library(twitteR)
library(httr)
library(sentiment)
library(DT)
library(wordcloud)
library(ggplot2) #used for generating graphs
library(rjson) #used for converting R objects to human readable text and vice versa
library(bigmemory) #used for managing massive amounts of data
library(biganalytics) #used for managing massive amounts of data
library(bigtabulate) #used for managing massive amounts of data
library(stringr)
library(jsonlite)
library(e1071) #used for performing SVM algorithm
library(caret)
library(RTextTools)
library(tm)
library(dplyr)
library(wordcloud)
```

# Execute below function to return scores of the tweets

```
score.sentiment = function(sentences, pos.words, neg.words, .progress='none')
{
```

```
require(plyr)
require(stringr)
 sentences = gsub('[[:punct:]]', '', sentences)
sentences = gsub('[[:cntrl:]]', '', sentences)
sentences = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', sentences)
sentences = gsub('@\\w+', '', sentences)
sentences = gsub('[[:punct:]]', '', sentences)
sentences = gsub('[[:digit:]]', '', sentences)
sentences = gsub('http\\w+', '', sentences)
sentences = gsub('[ \t]{2,}', '', sentences)
sentences = gsub('^\\s+|\\s+$', '', sentences)
sentences = gsub('\\d+', '', sentences)
try.error = function(x)
{
  y = NA
  try_error = tryCatch(tolower(x), error=function(e) e)
  if (!inherits(try_error, 'error'))
    y = tolower(x)
  return(y)
}
sentences = sapply(sentences, try.error)
sentences = sentences[!is.na(sentences)]
names(sentences) = NULL

sentences = tolower(sentences)

scores = laply(sentences, function(sentence, pos.words, neg.words) {

  sentence = gsub('[[:punct:]]', '', sentence)
  sentence = gsub('[[:cntrl:]]', '', sentence)
  sentence = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', sentence)
  sentence = gsub('@\\w+', '', sentence)
  sentence = gsub('[[:punct:]]', '', sentence)
  sentence = gsub('[[:digit:]]', '', sentence)
  sentence = gsub('http\\w+', '', sentence)
  sentence = gsub('[ \t]{2,}', '', sentence)
  sentence = gsub('^\\s+|\\s+$', '', sentence)
  sentence = gsub('\\d+', '', sentence)
  try.error = function(x)
  {
    y = NA
    try_error = tryCatch(tolower(x), error=function(e) e)
    if (!inherits(try_error, 'error'))
      y = tolower(x)
    return(y)
  }
  sentence = sapply(sentence, try.error)
  sentence = sentence[!is.na(sentence)]
  names(sentence) = NULL
  # and convert to lower case:
  sentence = tolower(sentence)

  # split into words. str_split is in the stringr package
  word.list = str_split(sentence,'\\s+')
  # sometimes a list() is one level of hierarchy too much
  words = unlist(word.list)

  # compare our words to the dictionaries of positive & negative terms
  pos.matches = match(words, pos.words)
  neg.matches = match(words, neg.words)

  # match() returns the position of the matched term or NA
```

```
    # we just want a TRUE/FALSE:
    pos.matches = !is.na(pos.matches)
    neg.matches = !is.na(neg.matches)

    # and conveniently enough, TRUE/FALSE will be treated as 1/0 by sum():
    score = sum(pos.matches) - sum(neg.matches)
    return(score)
  }, pos.words, neg.words, .progress=.progress )

  scores.df = data.frame(score=scores, text=sentences)
  return(scores.df)
}




api_key <- "t4f2zQp9044P92l5J6PCz9wW2"
api_secret <- "cEKCFf9YSRYMnqMnKXizZ4RKB8dixPdGXgdtYmLYrY1xYY4HAV"
access_token <- "2717578724-rzBS0rZcNBg6xecEpFukQTFP4DVd4adeOgrhGHT"
access_token_secret <- "MeUkekEJ2Hu5f8uD9YECWBMhjJ9YdQ3mC9H7jbcbd00l0"
setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)

#Step1: Tweets Extraction
pos.words <- read.csv(file.choose())
neg.words <- read.csv(file.choose())

pos.words <- scan(file.choose(),what = 'character')
neg.words <- scan(file.choose(),what = 'character')

pos.words = c(pos.words, 'new','nice' ,'good', 'horizon')
```

```
neg.words = c(neg.words, 'behind','feels', 'ugly', 'back','worse' , 'shitty', 'bad',
'no','freaking','sucks','horrible')

modi_tweets <- searchTwitter("$narendramodi", n = 1000,lang='en')
modi_tweets

require(plyr)
test <- ldply(modi_tweets,function(t) t$toDataFrame() )
result <- score.sentiment(test$text,pos.words,neg.words)
summary(result$score)
count(result$score)
qplot(result$score,xlab = "Score of tweets")

#store score of each tweet, polarity will be our classes for SVM and corresponding
tweets in a data frame
namo_group_df=data.frame(score=result$score,polarity= result$score, tweets= result$text)

# SVM defined classes Neutral = 0, Negative = -1 and Positive = 1
# score 0 is class 0
# score 1 and above is class 1
# score less than 0 is class -1
for (i in 1:NROW(namo_group_df)){
  eachScore= namo_group_df[i,"score"]
  if(eachScore == 0){
    namo_group_df[i,"polarity"] = "0"
  } else if(eachScore == 1){
    namo_group_df[i,"polarity"] = "1"
  } else if(eachScore == 2){
    namo_group_df[i,"polarity"] = "1"
  }
  else if(eachScore == 3){
    namo_group_df[i,"polarity"] = "1"
  }else if(eachScore == 4){
    namo_group_df[i,"polarity"] = "1"
  }
  else if(eachScore == -1){
    namo_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -2){
    namo_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -3){
    namo_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -4){
    namo_group_df[i,"polarity"] = "-1"
  }
}
qplot(namo_group_df$score, data=namo_group_df,color=namo_group_df$polarity)
x <- subset(namo_group_df, select=-namo_group_df$polarity)
y <- namo_group_df$polarity
namo_group_df
write.csv(namo_group_df,"NamoResults20.csv")
```

# SVM Classficiation starts

# SVM classification needs classes, training set and testing set

#partition the data into two sets where 70% of the data will be assigned as training set and rest is testing set

```
intrain <- createDataPartition(y = namo_group_df$polarity, p= 0.7, list = FALSE)
```

#below is training set

```
training <- namo_group_df[intrain,]
```

```
#below is testing set
testing <- namo_group_df[-intrain,]

trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

# polarity column in the namo_group_df is used for classes as mentioned above
# As there are three classes and one outlier which is -5 in the classes. SVM will have
two hyper planes
svm_Linear <- train(polarity ~., data = training, method = "svmLinear",
                 trControl=trctrl,
                 preProcess = c("center", "scale"),
                 tuneLength = 10)

# View SVM results
print(svm_Linear)

# Now using the SVM analysis predict the data using testing data
test_pred <- predict(svm_Linear, newdata = testing)

# view prediction results
# this is used for comparing the polarity generated using Bayes algoritm
plot(test_pred)


#confusion matrix
confusionMatrix(test_pred,testing$polarity)
```

5.3.2.2 DONALD TRUMP

```
library(devtools)
library(twitteR)
library(httr)
library(sentiment)
library(DT)
library(wordcloud)
library(ggplot2)
library(rjson)
library(bigmemory)
library(biganalytics)
library(bigtabulate)
library(stringr)
library(jsonlite)
library(e1071)
library(caret)
library(RTextTools)
library(tm)
library(dplyr)
library(wordcloud)
```

# Execute below function to return scores of the tweets

```
score.sentiment = function(sentences, pos.words, neg.words, .progress='none')
{
  require(plyr)
  require(stringr)
  sentences = gsub('[[:punct:]]', '', sentences)
  sentences = gsub('[[:cntrl:]]', '', sentences)
  sentences = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', sentences)
  sentences = gsub('@\\w+', '', sentences)
  sentences = gsub('[[:punct:]]', '', sentences)
  sentences = gsub('[[:digit:]]', '', sentences)
  sentences = gsub('http\\w+', '', sentences)
  sentences = gsub('[ \t]{2,}', '', sentences)
  sentences = gsub('^\\s+|\\s+$', '', sentences)
  sentences = gsub('\\d+', '', sentences)
  try.error = function(x)
  {
    y = NA
    try_error = tryCatch(tolower(x), error=function(e) e)
    if (!inherits(try_error, 'error'))
      y = tolower(x)
    return(y)
  }
  sentences = sapply(sentences, try.error)
  sentences = sentences[!is.na(sentences)]
  names(sentences) = NULL
   sentences = tolower(sentences)
    scores = laply(sentences, function(sentence, pos.words, neg.words) {
     sentence = gsub('[[:punct:]]', '', sentence)
    sentence = gsub('[[:cntrl:]]', '', sentence)
    sentence = gsub('(RT|via)((?:\\b\\W*@\\w+)+)', '', sentence)
    sentence = gsub('@\\w+', '', sentence)
    sentence = gsub('[[:punct:]]', '', sentence)
    sentence = gsub('[[:digit:]]', '', sentence)
    sentence = gsub('http\\w+', '', sentence)
    sentence = gsub('[ \t]{2,}', '', sentence)
    sentence = gsub('^\\s+|\\s+$', '', sentence)
    sentence = gsub('\\d+', '', sentence)
    try.error = function(x)
    {
      y = NA
      try_error = tryCatch(tolower(x), error=function(e) e)
      if (!inherits(try_error, 'error'))
```

```
      y = tolower(x)
     return(y)
    }
    sentence = sapply(sentence, try.error)
    sentence = sentence[!is.na(sentence)]
    names(sentence) = NULL
    # and convert to lower case:
    sentence = tolower(sentence)

    # split into words. str_split is in the stringr package
    word.list = str_split(sentence,'\\s+')
    # sometimes a list() is one level of hierarchy too much
    words = unlist(word.list)

    # compare our words to the dictionaries of positive & negative terms
    pos.matches = match(words, pos.words)
    neg.matches = match(words, neg.words)

    # match() returns the position of the matched term or NA
    # we just want a TRUE/FALSE:
    pos.matches = !is.na(pos.matches)
    neg.matches = !is.na(neg.matches)

    # and conveniently enough, TRUE/FALSE will be treated as 1/0 by sum():
    score = sum(pos.matches) - sum(neg.matches)
    return(score)
  }, pos.words, neg.words, .progress=.progress )

  scores.df = data.frame(score=scores, text=sentences)
  return(scores.df)
}
api_key <- "t4f2zQp9044P92l5J6PCz9wW2"
```

```
api_secret <- "cEKCFf9YSRYMnqMnKXizZ4RKB8dixPdGXgdtYmLYrY1xYY4HAV"
access_token <- "2717578724-rzBS0rZcNBg6xecEpFukQTFP4DVd4adeOgrhGHT"
access_token_secret <- "MeUkekEJ2Hu5f8uD9YECWBMhjJ9YdQ3mC9H7jbcbd00l0"
setup_twitter_oauth(api_key,api_secret,access_token,access_token_secret)
```

#Step1: Tweets Extraction

```
pos.words <- read.csv(file.choose())
neg.words <- read.csv(file.choose())

pos.words <- scan(file.choose(),what = 'character')
neg.words <- scan(file.choose(),what = 'character')

pos.words = c(pos.words, 'new','nice' ,'good', 'horizon')
neg.words = c(neg.words, 'behind','feels', 'ugly', 'back','worse' , 'shitty', 'bad', 'no','freaking','sucks','horrible')

trump_tweets <- searchTwitter("$donaldtrump", n = 1000,lang='en')
trump_tweets

require(plyr)
test <- ldply(trump_tweets,function(t) t$toDataFrame() )
result <- score.sentiment(test$text,pos.words,neg.words)
summary(result$score)
count(result$score)
qplot(result$score,xlab = "Score of tweets")
```

#store score of each tweet, polarity will be our classes for SVM and corresponding tweets in a data frame

```
dtump_group_df=data.frame(score=result$score,polarity= result$score, tweets= result$text)
```

# SVM defined classes Neutral = 0, Negative = -1 and Positive = 1

```
# score 0 is class 0
# score 1 and above is class 1
# score less than 0 is class -1
for (i in 1:NROW(dtrump_group_df)){
  eachScore= dtrump_group_df[i,"score"]
  if(eachScore == 0){
    dtrump_group_df[i,"polarity"] = "0"
  } else if(eachScore == 1){
    dtrump_group_df[i,"polarity"] = "1"
  } else if(eachScore == 2){
    dtrump_group_df[i,"polarity"] = "1"
  }
  else if(eachScore == 3){
    dtrump_group_df[i,"polarity"] = "1"
  }else if(eachScore == 4){
    dtrump_group_df[i,"polarity"] = "1"
  }
  else if(eachScore == -1){
    dtrump_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -2){
    dtrump_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -3){
    dtrump_group_df[i,"polarity"] = "-1"
  }
  else if(eachScore == -4){
    dtrump_group_df[i,"polarity"] = "-1"
  }
}
qplot(dtrump_group_df$score, data=dtrump_group_df,color=dtrump_group_df$polarity)
```

```
x <- subset(dtrump_group_df, select=-dtrump_group_df$polarity)
y <- dtrump_group_df$polarity
dtrump_group_df
write.csv(dtrump_group_df,"DtrumpResults20.csv")

# SVM Classficiation starts
# SVM classification needs classes, training set and testing set

#partition the data into two sets where 70% of the data will be assigned as training set and rest is testing set
intrain <- createDataPartition(y = dtrump_group_df$polarity, p= 0.7, list = FALSE)
#below is training set
training <- dtrump_group_df[intrain,]
#below is testing set
testing <- dtrump_group_df[-intrain,]

trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

# polarity column in the namo_group_df is used for classes as mentioned above
# As there are three classes and one outlier which is -5 in the classes. SVM will have two hyper planes
svm_Linear <- train(polarity ~., data = training, method = "svmLinear",
                trControl=trctrl,
                preProcess = c("center", "scale"),
                tuneLength = 10)

# View SVM results
print(svm_Linear)

# Now using the SVM analysis predict the data using testing data
test_pred <- predict(svm_Linear, newdata = testing)
```

# view prediction results

# this is used for comparing the polarity generated using Bayes algoritm

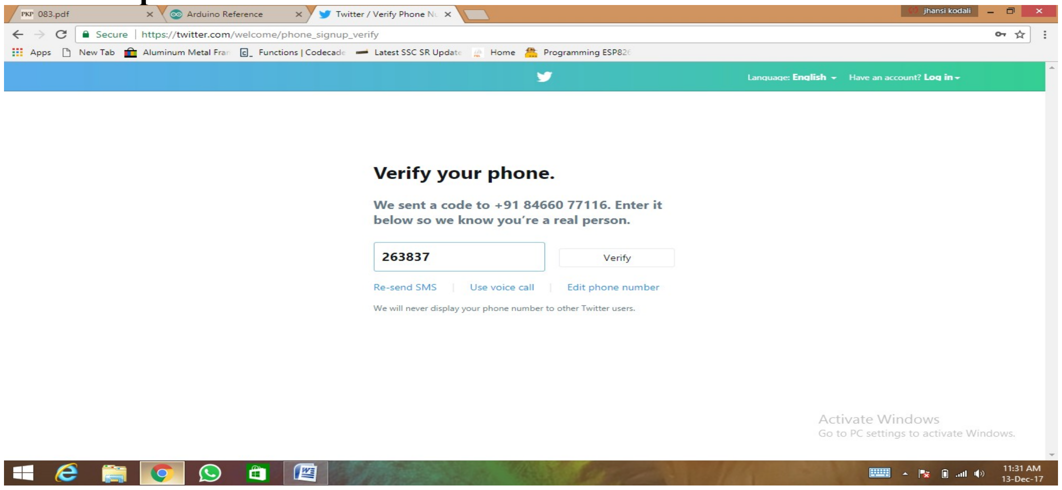
plot(test_pred)

#confusion matrix

confusionMatrix(test_pred,testing$polarity)

5.4 OUTPUT SCREENS

5.4.1 Output Screens Twitter Creation



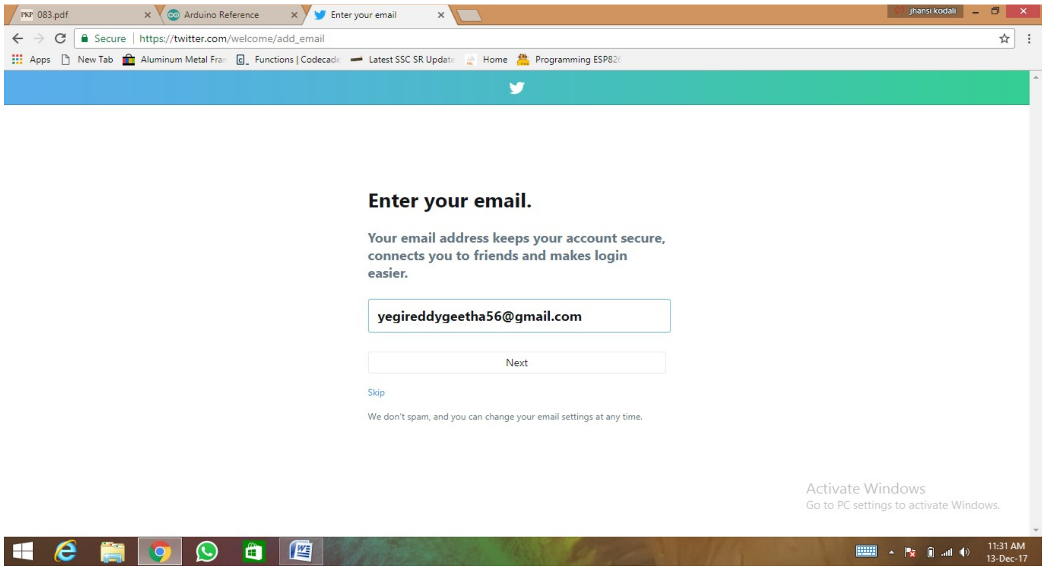
Verify your phone.

We sent a code to +91 84660 77116. Enter it below so we know you're a real person.

263837

We will never display your phone number to other Twitter users.

Fig 5.1 : Phone number verification


https://twitter.com/welcome/add_email
Next
Skip
Activate Windows
Go to PC settings to activate Windows.

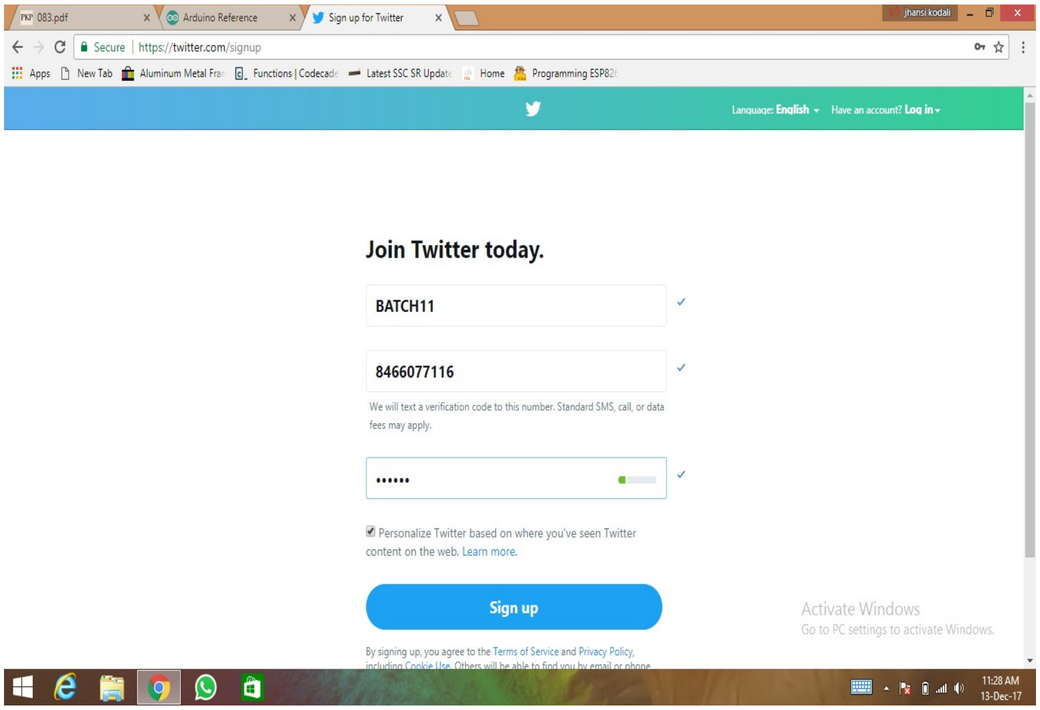
Enter your email.

Your email address keeps your account secure, connects you to friends and makes login easier.

yegireddygeetha56@gmail.com

We don't spam, and you can change your email settings at any time.

Fig 5.2 : Email verification



Join Twitter today.

BATCH11

8466077116

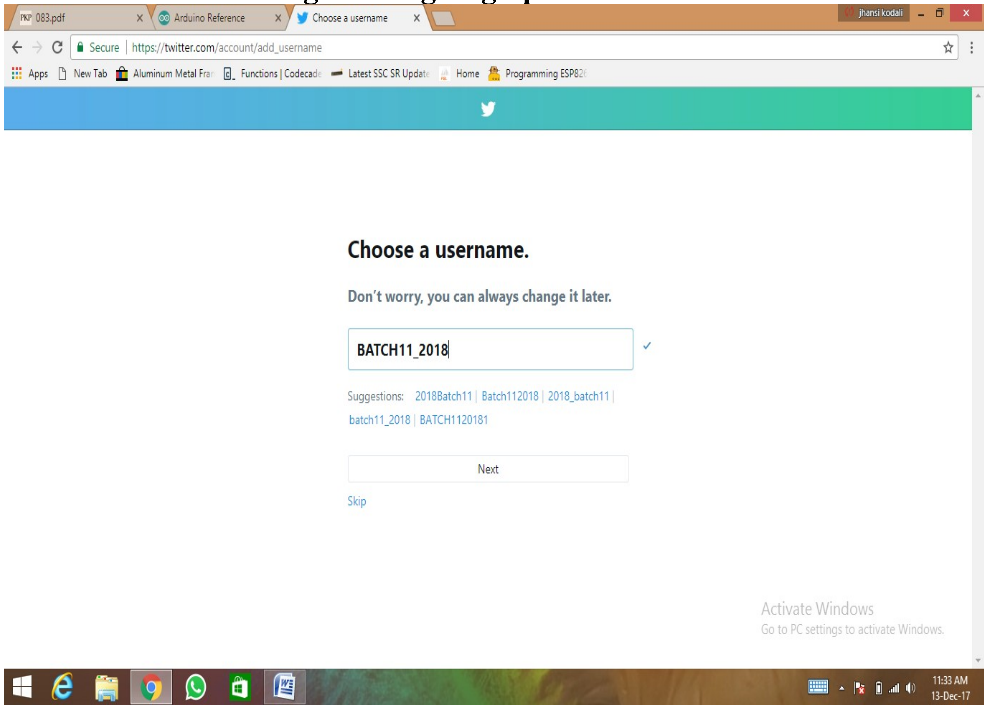
We will text a verification code to this number. Standard SMS, call, or data fees may apply.

••••••

Personalize Twitter based on where you've seen Twitter content on the web. Learn more.

By signing up, you agree to the Terms of Service and Privacy Policy, including Cookie Use. Others will be able to find you by email or phone

Fig 5.3 : Signing up into Twitter Account



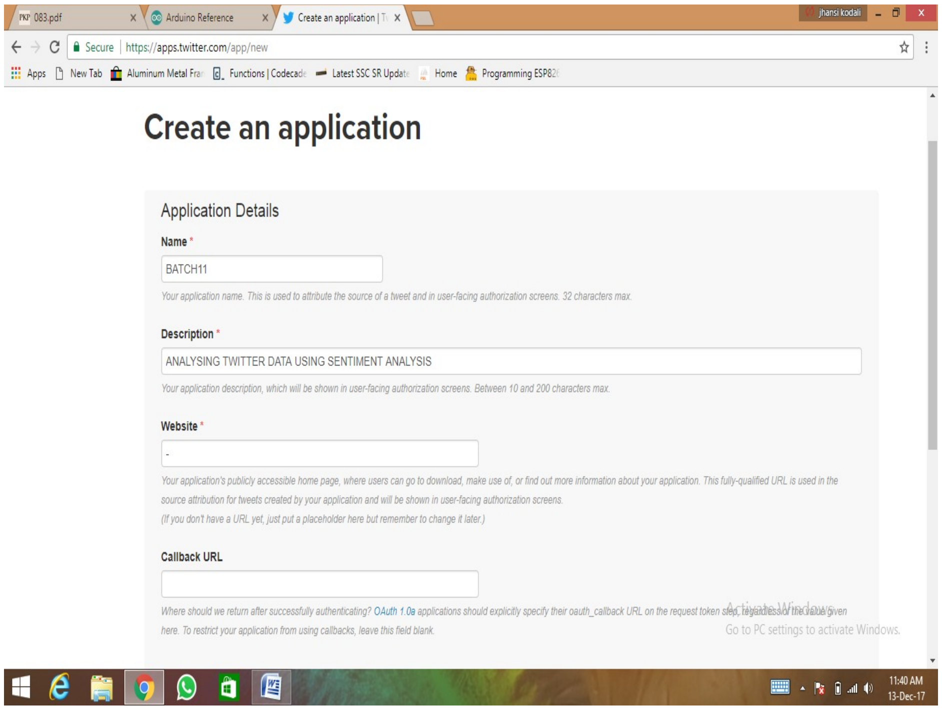
Choose a username.

Don't worry, you can always change it later.

BATCH11_2018

Suggestions: 2018Batch11 | Batch112018 | 2018_batch11 | batch11_2018 | BATCH1120181

Fig 5.4: Choosing the username



Create an application

Application Details

Name *

BATCH11

Your application name. This is used to attribute the source of a tweet and in user-facing authorization screens. 32 characters max.

Description *

ANALYSING TWITTER DATA USING SENTIMENT ANALYSIS

Your application description, which will be shown in user-facing authorization screens. Between 10 and 200 characters max.

Website *

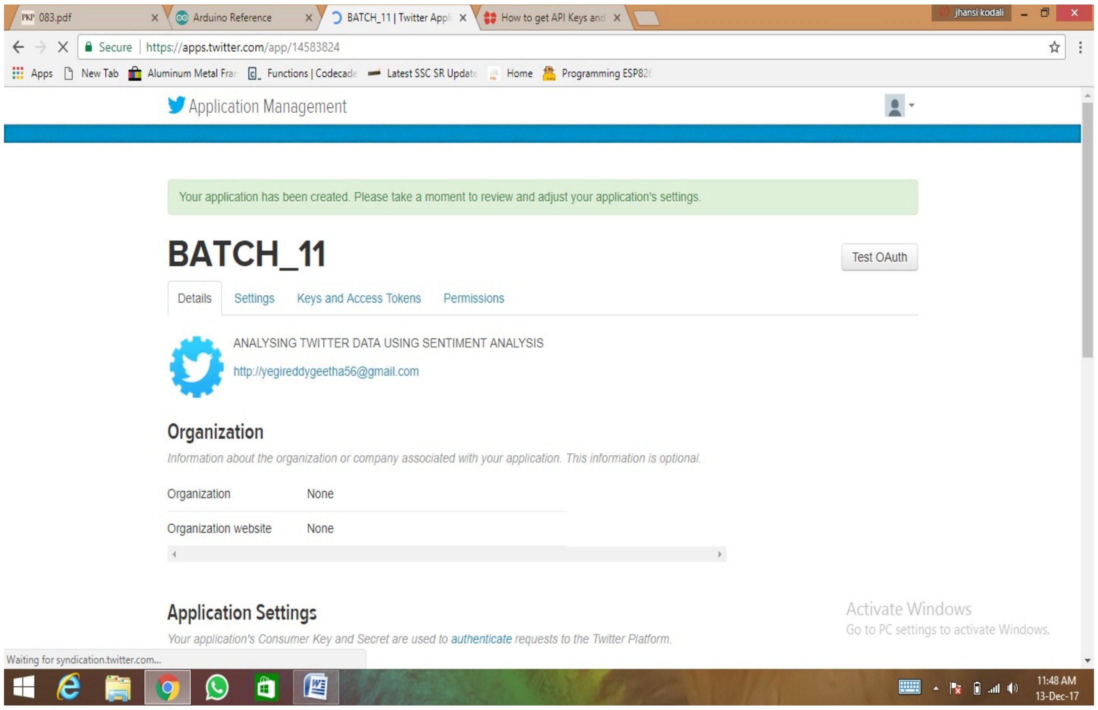
-

Your application's publicly accessible home page, where users can go to download, make use of, or find out more information about your application. This fully-qualified URL is used in the source attribution for tweets created by your application and will be shown in user-facing authorization screens.
(If you don't have a URL yet, just put a placeholder here but remember to change it later.)

Callback URL

Where should we return after successfully authenticating? OAuth 1.0a applications should explicitly specify their oauth_callback URL on the request token step, regardless of the value given here. To restrict your application from using callbacks, leave this field blank.

Fig5.5 : Creating a Twitter Application



Application Management

Your application has been created. Please take a moment to review and adjust your application's settings.

BATCH_11

ANALYSING TWITTER DATA USING SENTIMENT ANALYSIS

http://yegireddygeetha56@gmail.com

Organization

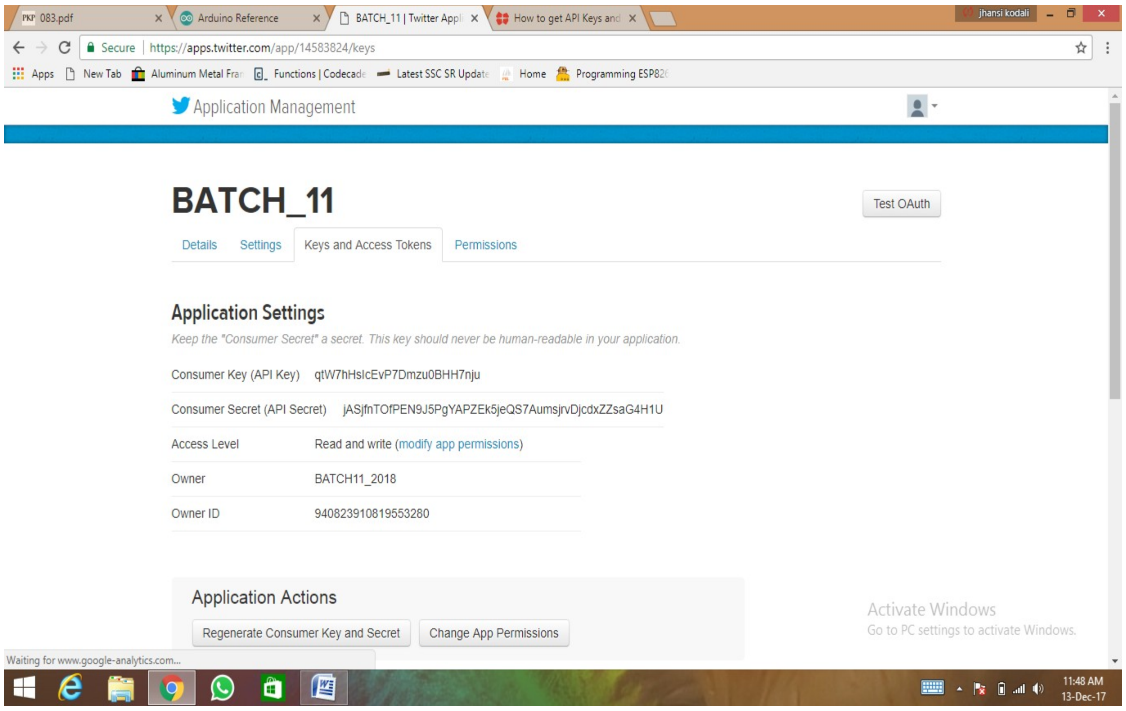
Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

Application Settings

Your application's Consumer Key and Secret are used to authenticate requests to the Twitter Platform.

Fig5.6 :Successful Creation Of Twitter Application



Application Management

BATCH_11

Details | Settings | Keys and Access Tokens | Permissions

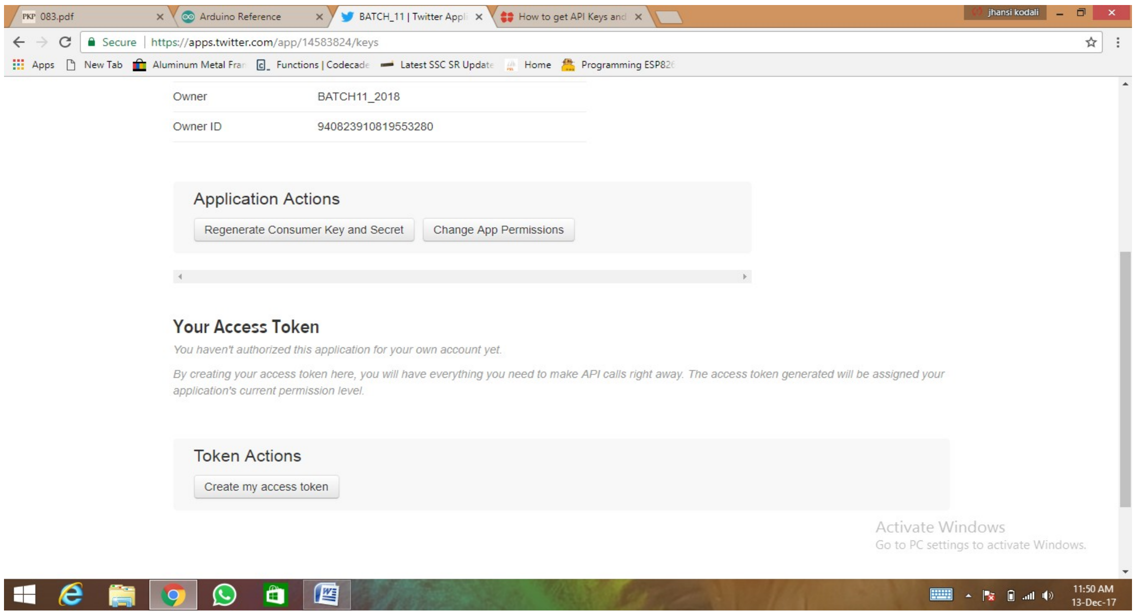
Application Settings

Keep the "Consumer Secret" a secret. This key should never be human-readable in your application.

Consumer Key (API Key)	qtW7hHslcEvP7Dmzu0BHH7nju
Consumer Secret (API Secret)	jASjfnTOfPEN9J5PgYAPZEk5jeQS7AumsjrvDjcdxZZsaG4H1U
Access Level	Read and write (modify app permissions)
Owner	BATCH11_2018
Owner ID	940823910819553280

Application Actions

Fig5.7: Viewing Access Keys



Owner	BATCH11_2018
Owner ID	940823910819553280

Application Actions

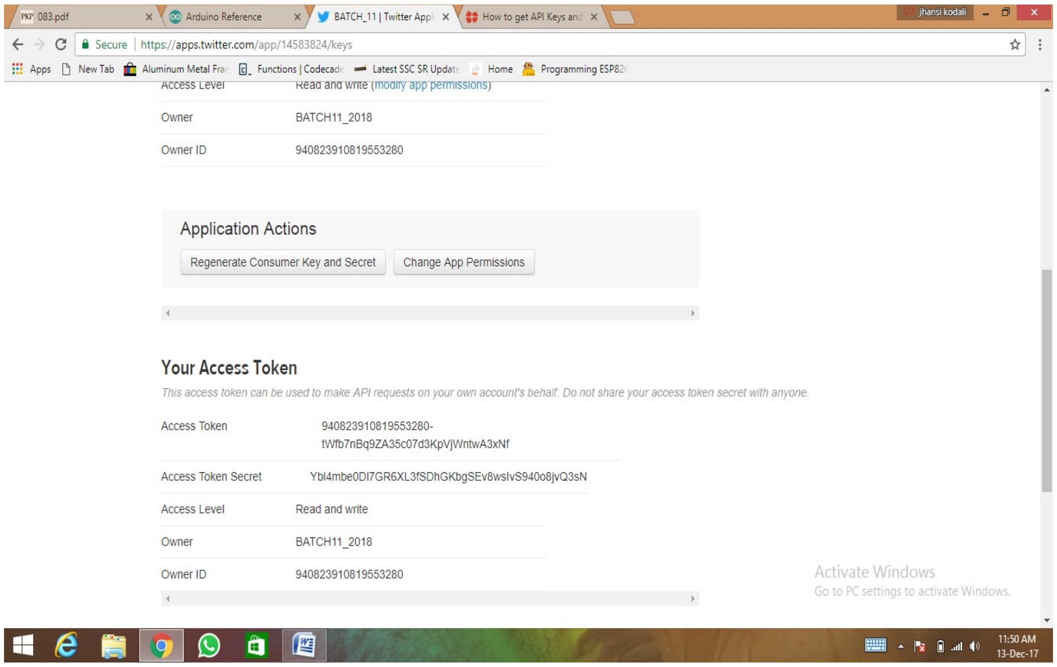
Your Access Token

You haven't authorized this application for your own account yet.

By creating your access token here, you will have everything you need to make API calls right away. The access token generated will be assigned your application's current permission level.

Token Actions

Fig5.8 : Creating Access Tokens



https://apps.twitter.com/app/14583824/keys

Access Level	Read and write (modify app permissions)
Owner	BATCH11_2018
Owner ID	940823910819553280

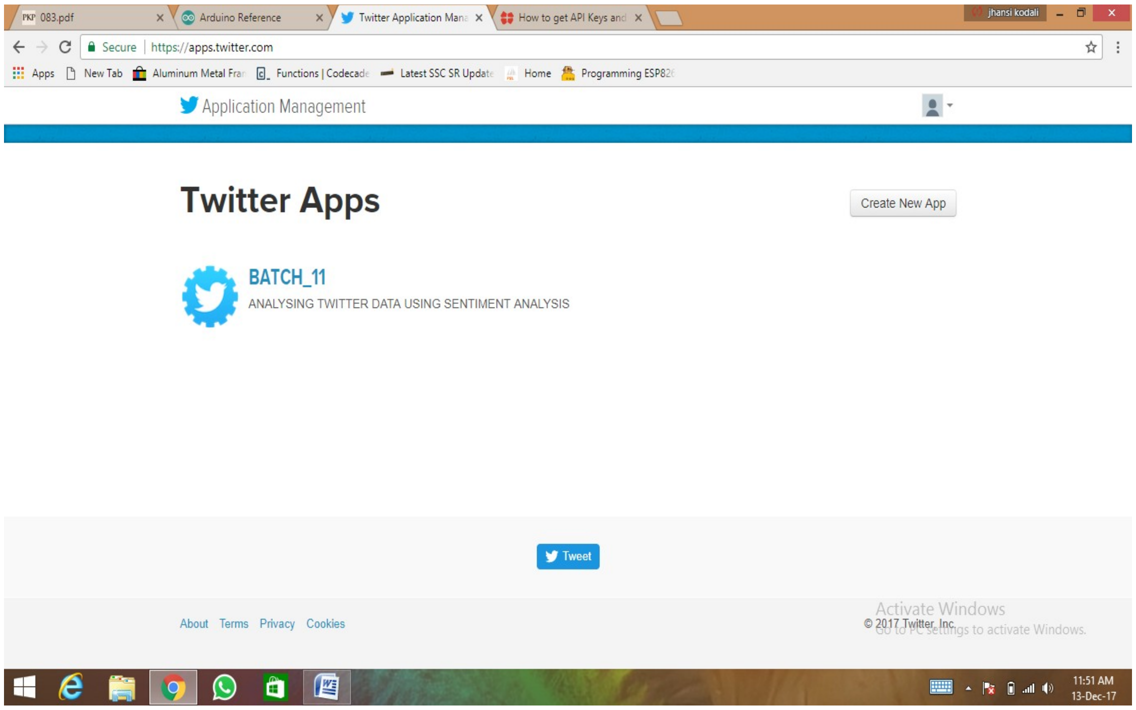
Application Actions

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	940823910819553280-tWfb7nBq9ZA35c07d3KpVjWntwA3xNf
Access Token Secret	Ybl4mbe0DI7GR6XL3fSDhGKbgSEv8wslvS940o8jvQ3sN
Access Level	Read and write
Owner	BATCH11_2018
Owner ID	940823910819553280

Fig5.9 :Viewing the Access Tokens



https://apps.twitter.com

Application Management

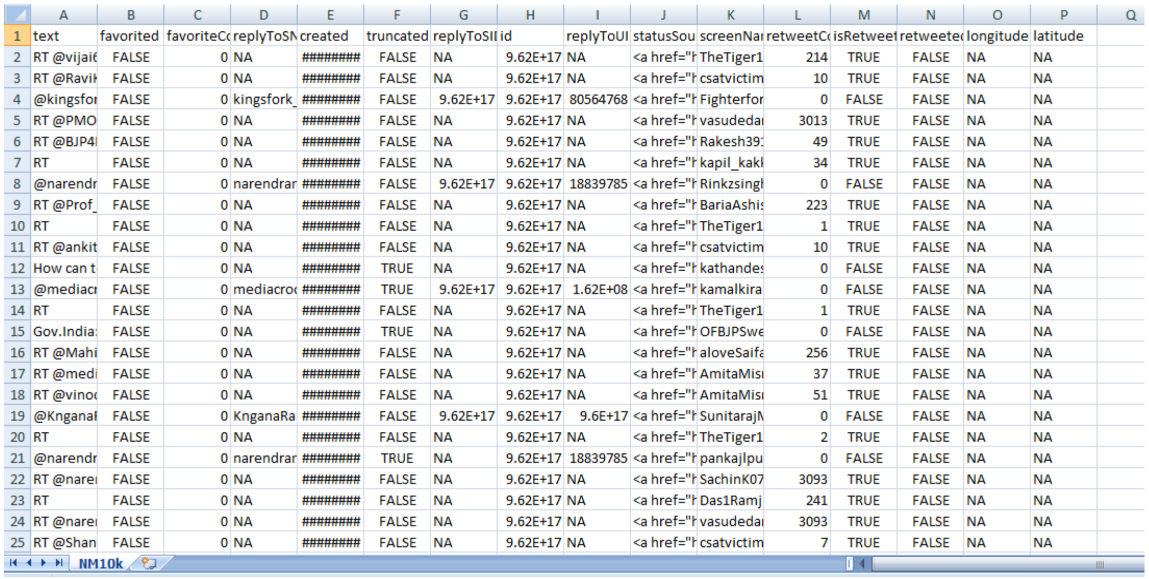
Twitter Apps

BATCH_11

ANALYSING TWITTER DATA USING SENTIMENT ANALYSIS

Fig5.10 : Successful Creation of Twitter Application

5.2.3 Output Screen of Narendra Modi & Donald Trump using Naïve Bayes

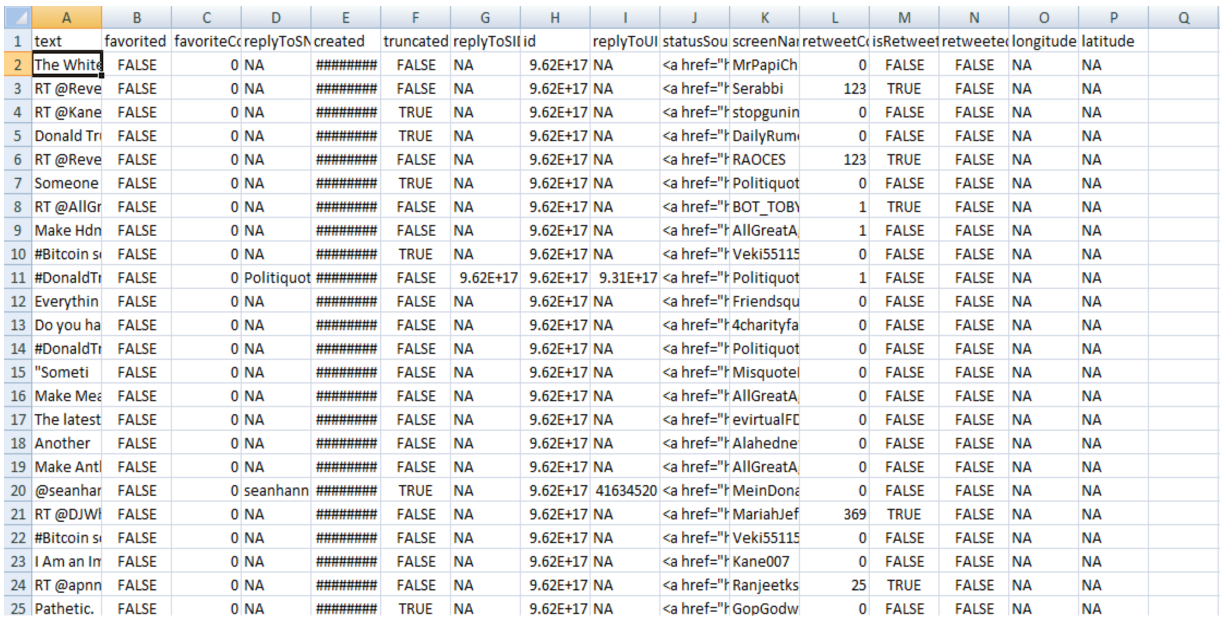


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	text	favorited	favoriteC	replyToSN	created	truncated	replyToSI	id	replyToUI	statusSou	screenNa	retweetC	isRetweet	retweete	longitude	latitude	
2	RT @vijai6	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	TheTiger1	214	TRUE	FALSE	NA	NA	
3	RT @Ravik	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	csatvictim	10	TRUE	FALSE	NA	NA	
4	@kingsfor	FALSE	0	kingsfork_	#########	FALSE	9.62E+17	9.62E+17	80564768	<a href="h	Fighterfor	0	FALSE	FALSE	NA	NA	
5	RT @PMO	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	vasudeda	3013	TRUE	FALSE	NA	NA	
6	RT @BJP4I	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Rakesh39	49	TRUE	FALSE	NA	NA	
7	RT	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	kapil_kak	34	TRUE	FALSE	NA	NA	
8	@narendr	FALSE	0	narendrar	#########	FALSE	9.62E+17	9.62E+17	18839785	<a href="h	Rinkzsingl	0	FALSE	FALSE	NA	NA	
9	RT @Prof_	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	BariaAshis	223	TRUE	FALSE	NA	NA	
10	RT	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	TheTiger1	1	TRUE	FALSE	NA	NA	
11	RT @ankit	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	csatvictim	10	TRUE	FALSE	NA	NA	
12	How can t	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	kathandes	0	FALSE	FALSE	NA	NA	
13	@mediacr	FALSE	0	mediacro	#########	TRUE	9.62E+17	9.62E+17	1.62E+08	<a href="h	kamalkira	0	FALSE	FALSE	NA	NA	
14	RT	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	TheTiger1	1	TRUE	FALSE	NA	NA	
15	Gov.India:	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	OFBJPSwe	0	FALSE	FALSE	NA	NA	
16	RT @Mahi	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	aloveSaifa	256	TRUE	FALSE	NA	NA	
17	RT @medi	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	AmitaMis	37	TRUE	FALSE	NA	NA	
18	RT @vinoc	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	AmitaMis	51	TRUE	FALSE	NA	NA	
19	@Knganal	FALSE	0	KnganaRa	#########	FALSE	9.62E+17	9.62E+17	9.6E+17	<a href="h	SunitarajN	0	FALSE	FALSE	NA	NA	
20	RT	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	TheTiger1	2	TRUE	FALSE	NA	NA	
21	@narendr	FALSE	0	narendrar	#########	TRUE	NA	9.62E+17	18839785	<a href="h	pankajlpu	0	FALSE	FALSE	NA	NA	
22	RT @narei	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	SachinK07	3093	TRUE	FALSE	NA	NA	
23	RT	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Das1Ramj	241	TRUE	FALSE	NA	NA	
24	RT @narei	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	vasudeda	3093	TRUE	FALSE	NA	NA	
25	RT @Shan	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	csatvictim	7	TRUE	FALSE	NA	NA	

NM10k

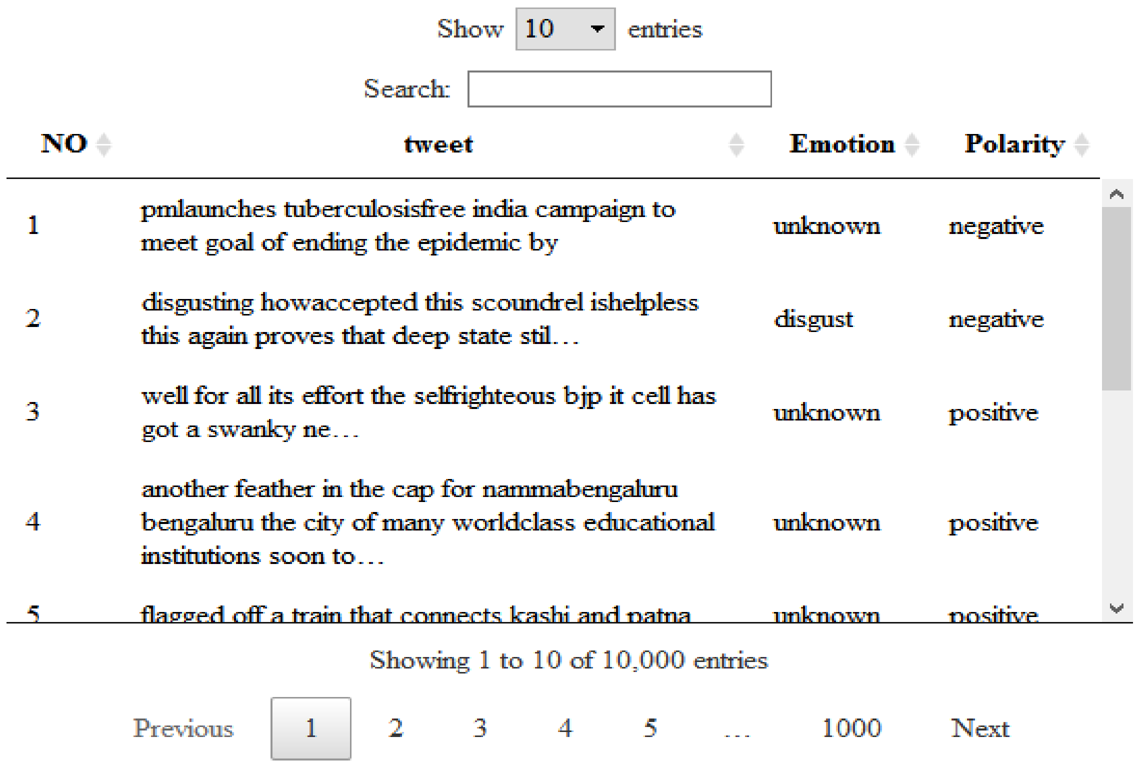
Fig5.11 Data set of Narendra Modi

This is the raw data set that has been extracted directly from twitter. A total of 10,000 tweets have been collected.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	text	favorited	favoriteC	replyToSN	created	truncated	replyToSI	id	replyToUI	statusSou	screenNa	retweetC	isRetweet	retweete	longitude	latitude	
2	The White	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	MrPapiCh	0	FALSE	FALSE	NA	NA	
3	RT @Reve	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Serabbi	123	TRUE	FALSE	NA	NA	
4	RT @Kane	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	stopgunin	0	FALSE	FALSE	NA	NA	
5	Donald Tr	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	DailyRum	0	FALSE	FALSE	NA	NA	
6	RT @Reve	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	RAOCES	123	TRUE	FALSE	NA	NA	
7	Someone	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	Politiquot	0	FALSE	FALSE	NA	NA	
8	RT @AllGr	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	BOT_TOBY	1	TRUE	FALSE	NA	NA	
9	Make Hdn	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	AllGreatA	1	FALSE	FALSE	NA	NA	
10	#Bitcoin s	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	Veki55115	0	FALSE	FALSE	NA	NA	
11	#DonaldTr	FALSE	0	Politiquot	#########	FALSE	9.62E+17	9.62E+17	9.31E+17	<a href="h	Politiquot	1	FALSE	FALSE	NA	NA	
12	Everythin	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Friendsqu	0	FALSE	FALSE	NA	NA	
13	Do you ha	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	4charityfa	0	FALSE	FALSE	NA	NA	
14	#DonaldTr	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Politiquot	0	FALSE	FALSE	NA	NA	
15	"Someti	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Misquotel	0	FALSE	FALSE	NA	NA	
16	Make Mea	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	AllGreatA	0	FALSE	FALSE	NA	NA	
17	The latest	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	evirtualFD	0	FALSE	FALSE	NA	NA	
18	Another	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Alahedne	0	FALSE	FALSE	NA	NA	
19	Make Antl	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	AllGreatA	0	FALSE	FALSE	NA	NA	
20	@seanhar	FALSE	0	seanhann	#########	TRUE	NA	9.62E+17	41634520	<a href="h	MeinDona	0	FALSE	FALSE	NA	NA	
21	RT @DJWI	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	MariahJef	369	TRUE	FALSE	NA	NA	
22	#Bitcoin s	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Veki55115	0	FALSE	FALSE	NA	NA	
23	I Am an In	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Kane007	0	FALSE	FALSE	NA	NA	
24	RT @apnn	FALSE	0	NA	#########	FALSE	NA	9.62E+17	NA	<a href="h	Ranjeetks	25	TRUE	FALSE	NA	NA	
25	Pathetic.	FALSE	0	NA	#########	TRUE	NA	9.62E+17	NA	<a href="h	GopGodw	0	FALSE	FALSE	NA	NA	

Fig5.12 Data set of Donald Trump



Show 10 entries

Search:

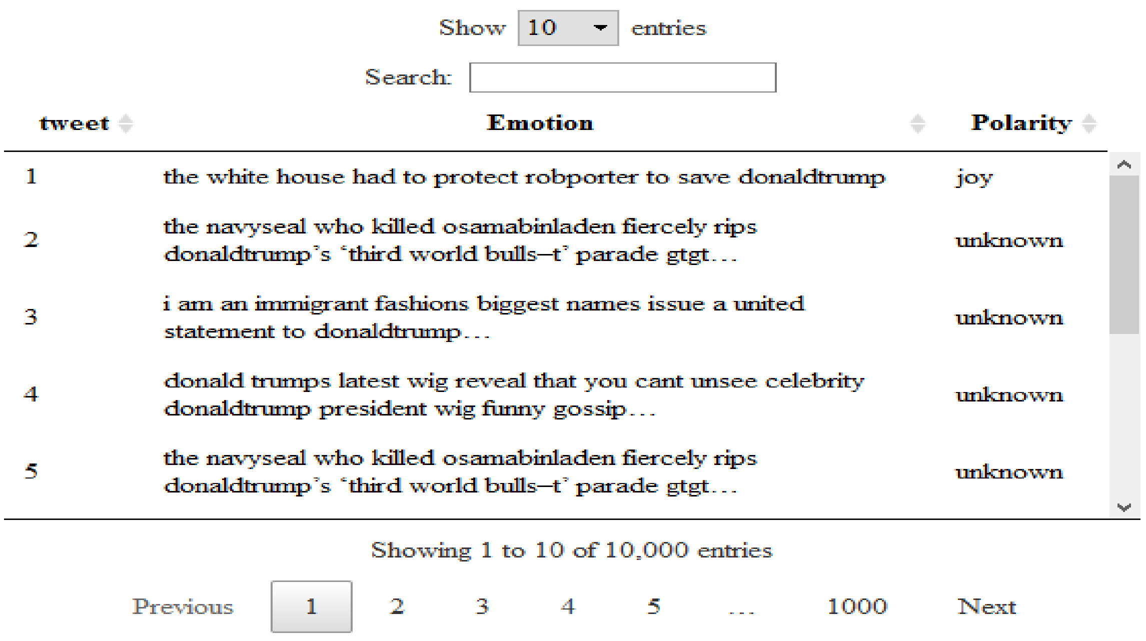
NO	tweet	Emotion	Polarity
1	pmlaunches tuberculosisfree india campaign to meet goal of ending the epidemic by	unknown	negative
2	disgusting howaccepted this scoundrel ishelpless this again proves that deep state stil…	disgust	negative
3	well for all its effort the selfrighteous bjp it cell has got a swanky ne…	unknown	positive
4	another feather in the cap for nammabengaluru bengaluru the city of many worldclass educational institutions soon to…	unknown	positive
5	flagged off a train that connects kashi and patna	unknown	positive

Showing 1 to 10 of 10,000 entries

Previous 1 2 3 4 5 … 1000 Next

Fig5.13 Data Table for Narendra Modi

It is the representation of tweets of Narendra Modi in a series with serial numbers. Here we classify the emotions and polarity of the tweets. This data table represents 10,000 tweets. The show entries option helps in displaying the number of tweets we select. We can also search for one particular tweet.


Show 10 entries

Search:

tweet	Emotion	Polarity
1	the white house had to protect robporter to save donaldtrump	joy
2	the navyseal who killed osamabinladen fiercely rips donaldtrump's 'third world bulls–t' parade gtgt…	unknown
3	i am an immigrant fashions biggest names issue a united statement to donaldtrump…	unknown
4	donald trumps latest wig reveal that you cant unsee celebrity donaldtrump president wig funny gossip…	unknown
5	the navyseal who killed osamabinladen fiercely rips donaldtrump's 'third world bulls–t' parade gtgt…	unknown

Showing 1 to 10 of 10,000 entries

Previous 1 2 3 4 5 … 1000 Next

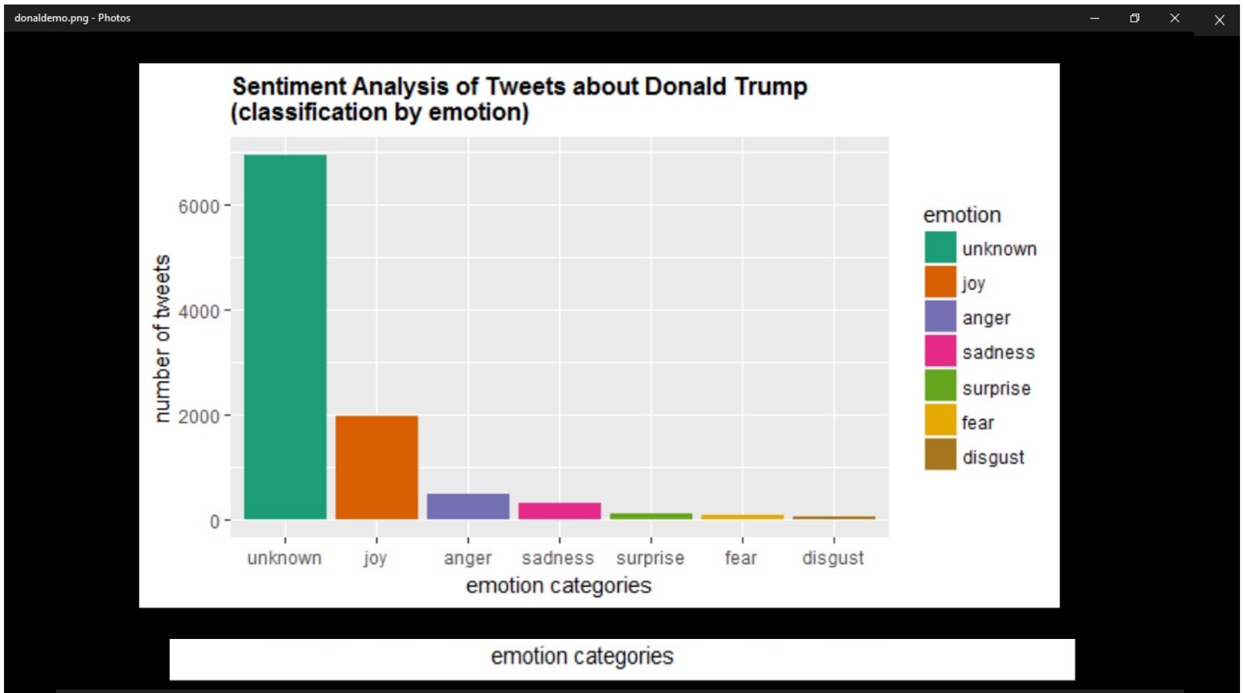
Fig: 5.14 Data Table

It is the representation of tweets of Donald Trump in a series with serial numbers. Here we classify the emotions and polarity of the tweets. This data table represents 10,000 tweets. The show entries option helps in displaying the number of tweets we select. We can also search for one particular tweet.

Fig5.15: Emotion Graph for Narendra Modi

The emotion graph displays six types of emotions. They are joy, sadness, surprise, anger, fear and disgust. On the y axis we have number of tweets and on the x axis we have categories of emotions. Unknown represents that those tweets did not fall into any category.

Fig5.16: Emotion Graph for Donald Trump



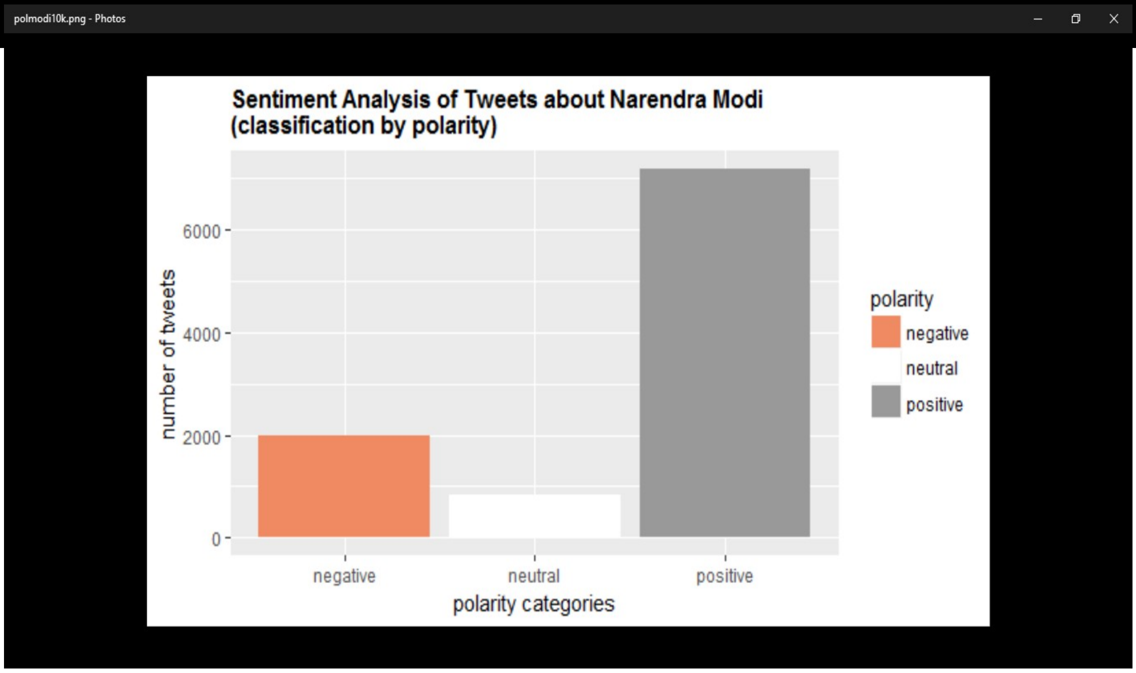


Fig 5.17 Polarity Graph for Narendra Modi

Polarity graph represents whether the tweet is positive , negative or neutral.

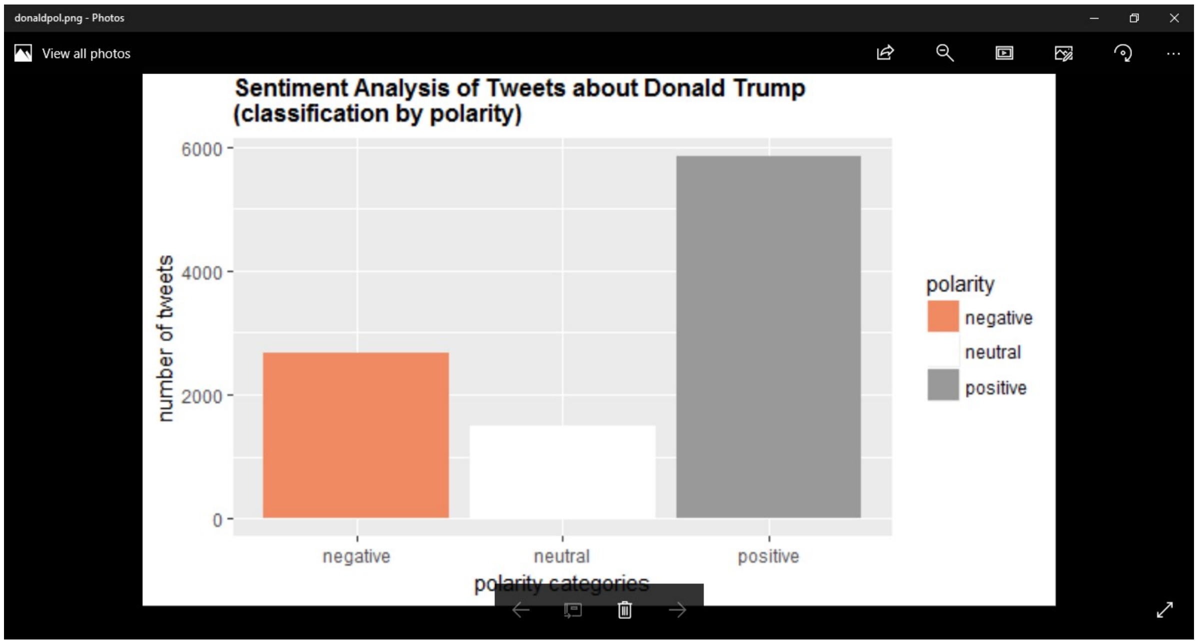
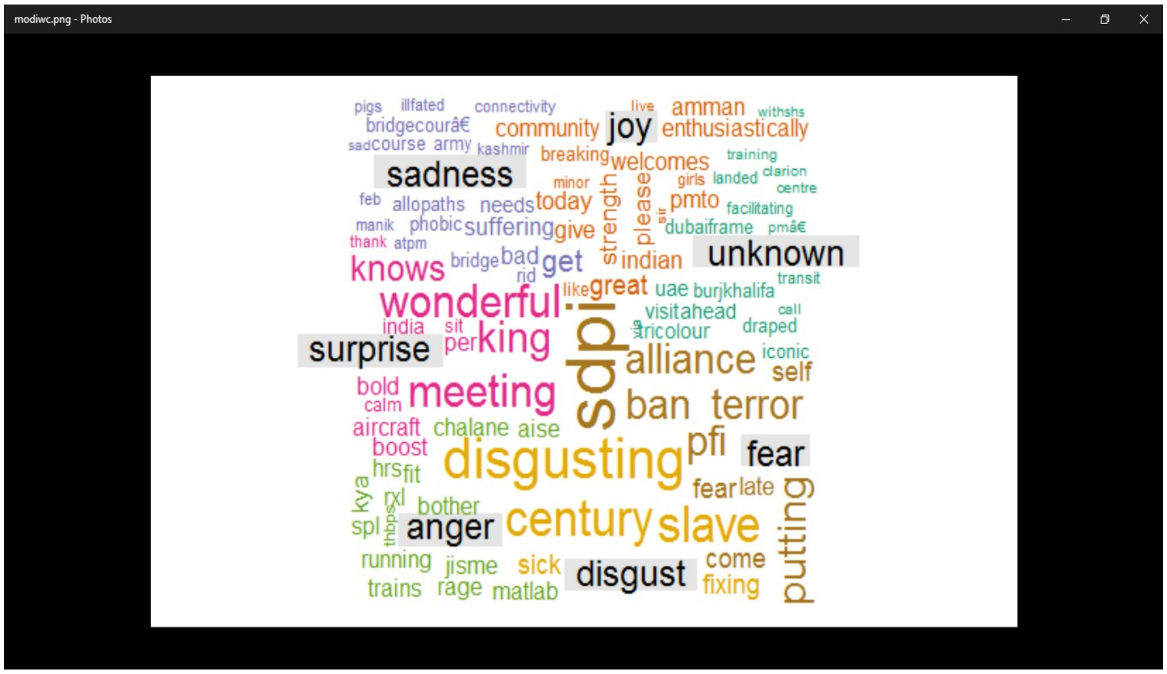
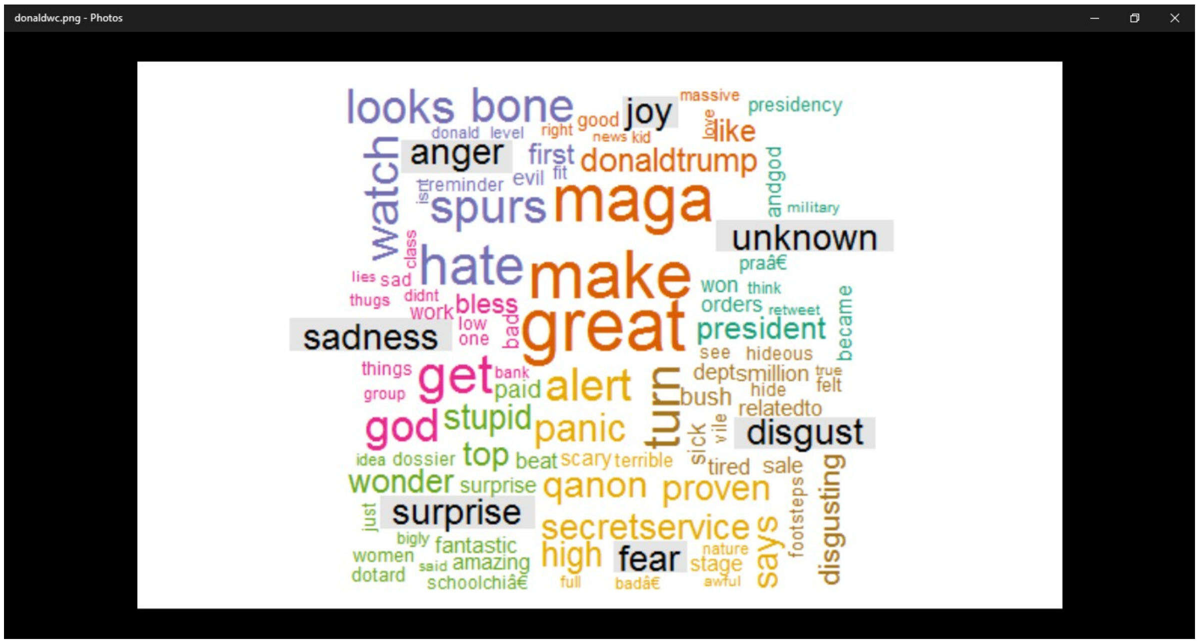


Fig 5.18 Polarity Graph for Donald trump



5.19 Word Cloud for Narendra Modi

The word cloud contains all the words from the tweets under their particular emotion. Greater the size of the word more the number of times it has been repeated.


donaldwc.png - Photos
looks bone good joy massive presidency
donald level right news kid love like
anger first donaldtrump
watch isn't reminder evil fit andgod
spurs maga military
unknown
class praâ€
hate make
lies sad won think
thugs didnt bless orders retweet
work great president became
sadness low one bad
see hideous
things get bank depts million true
group paid alert turn hide felt bush
relatedto
god stupid panic sick vile disgust
idea dossier top beat scary terrible tired sale
wonder surprise qanon proven
just surprise footsteps disgusting
bigly fantastic secretservice says
women said amazing high fear nature stage
dotard schoolchiâ€ full badâ€ awful

5.20 Word Cloud for Donald Trump

5.2.3 Output Screen Of Integrated result of Emotions and polarity:

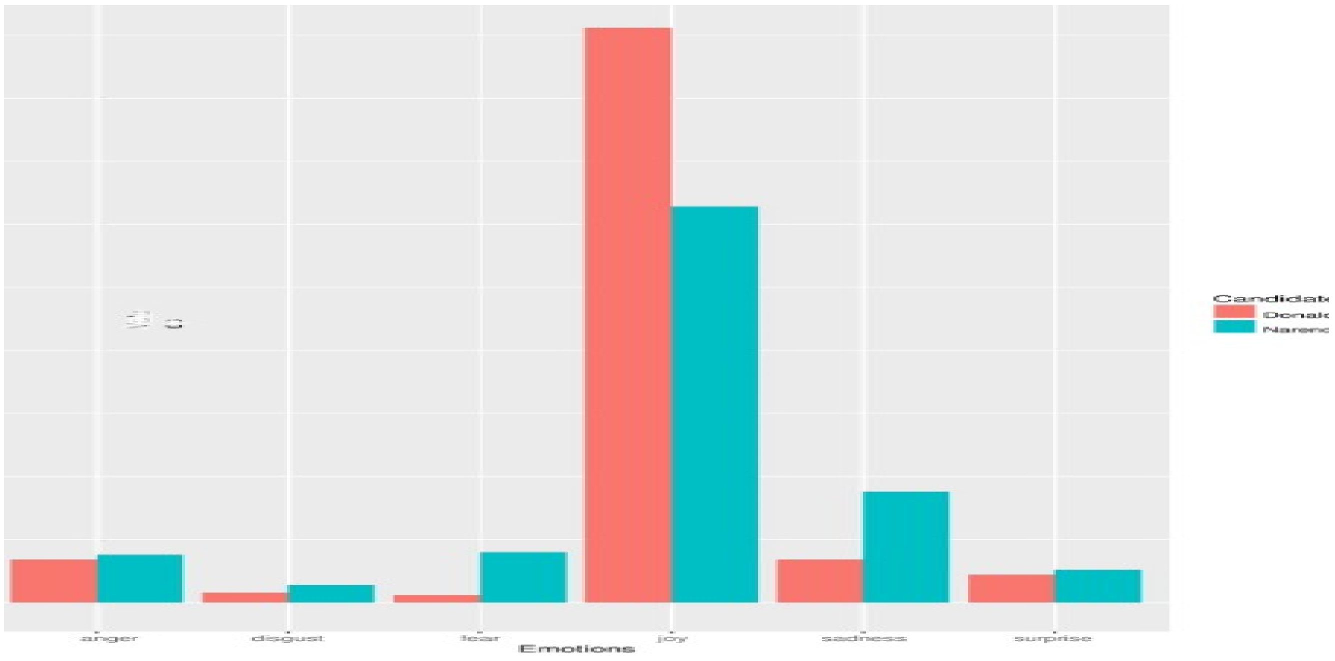


FIG5.21 Naïve bayes algotithm based integrated graph for emotion

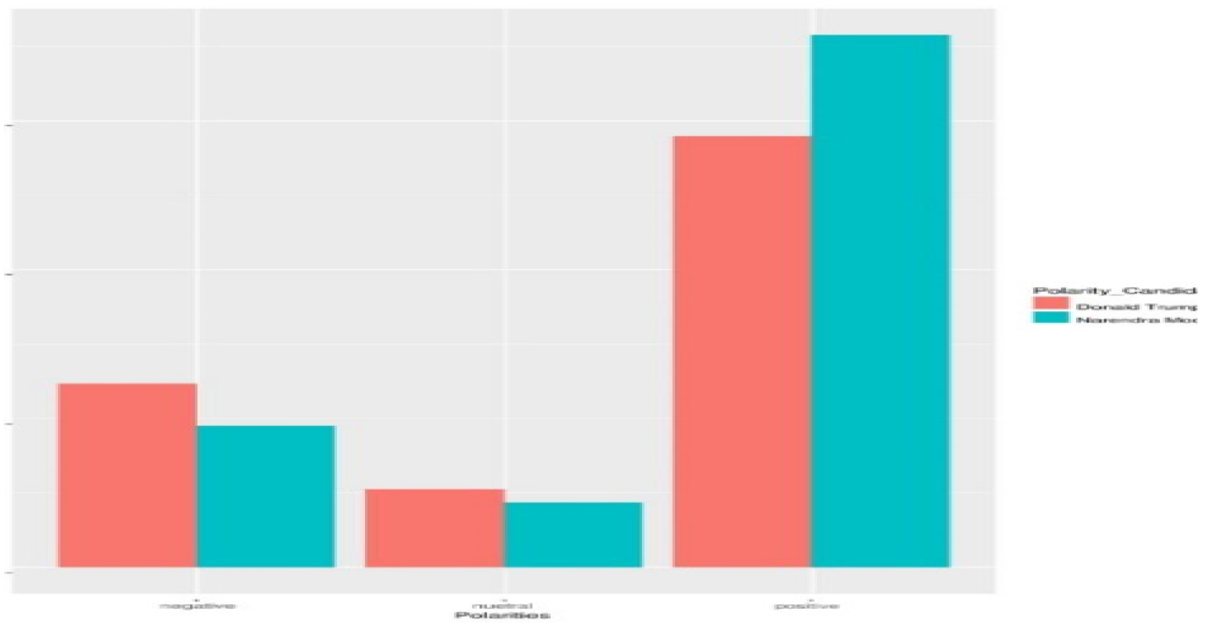


FIG 5.22 Naïve Bayes algorithm based integrated graph for Polarity

Output Screen of SVM

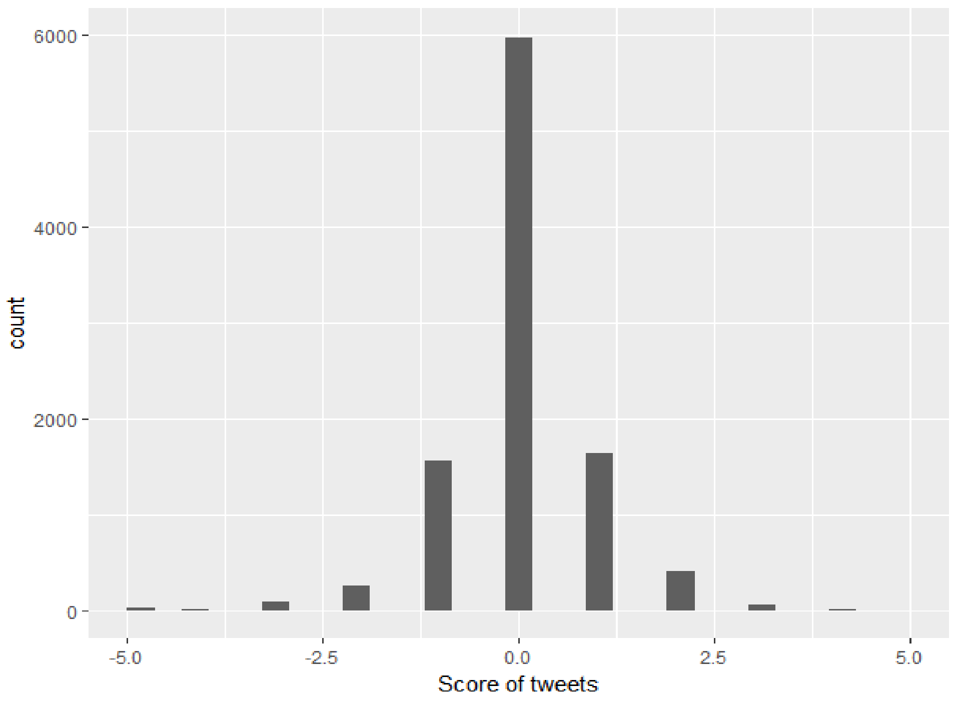


Fig 5.23 SVM scores of Narendra Modi

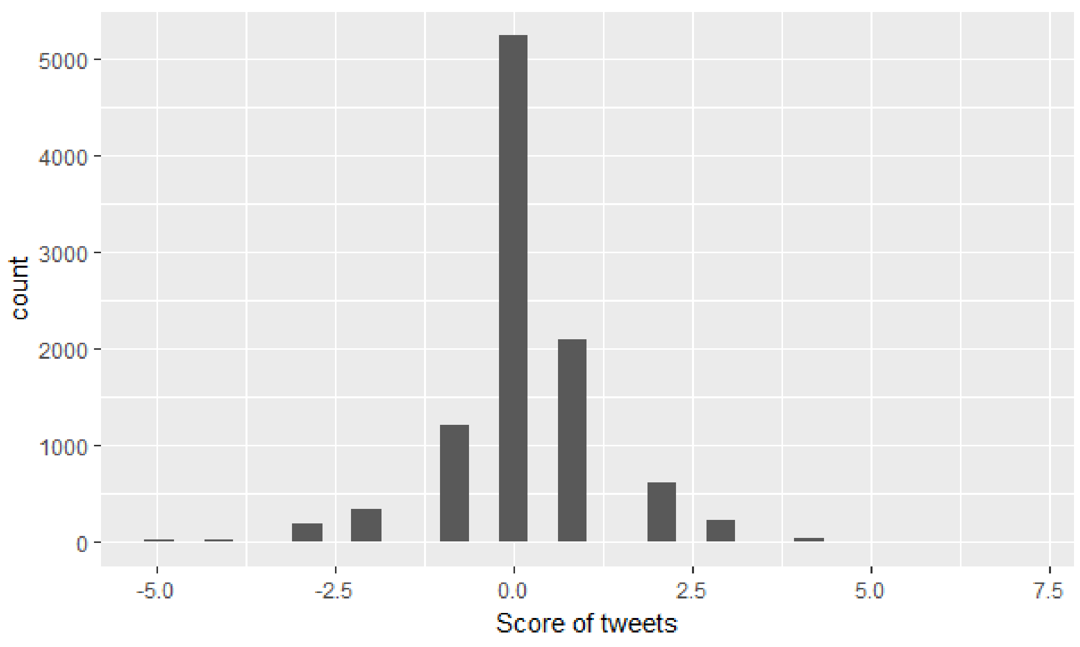
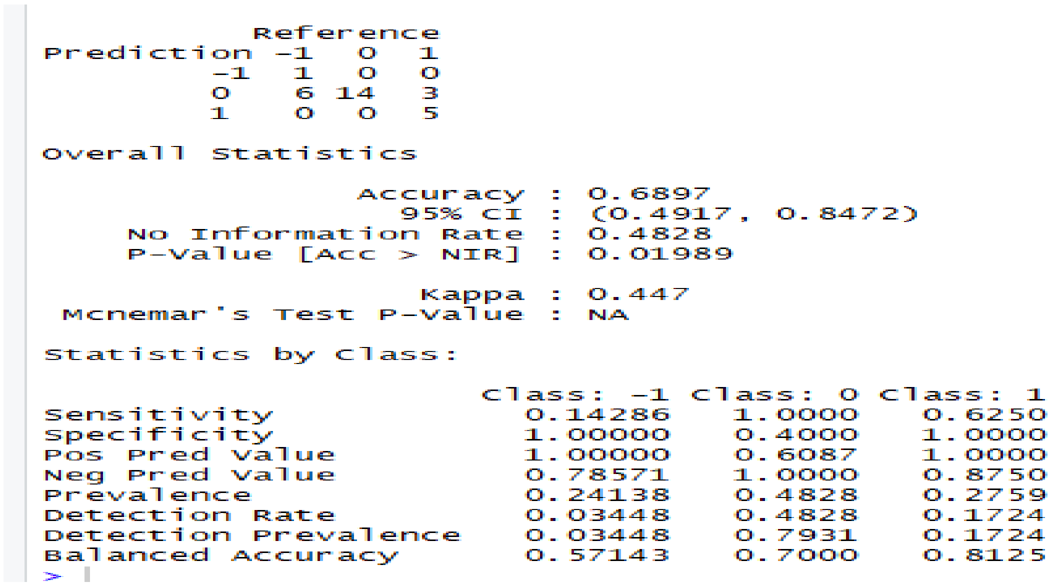
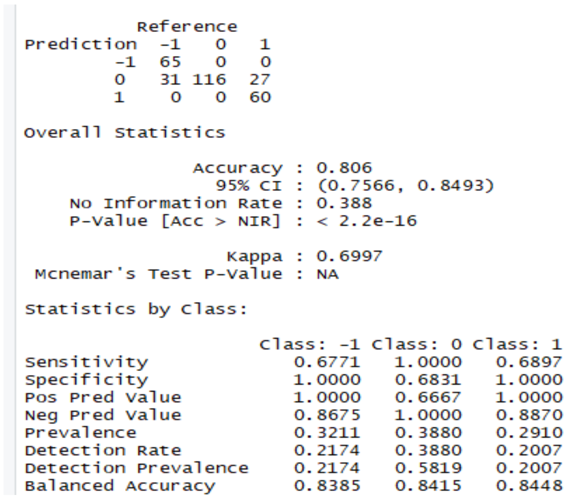


Fig: 5.24 SVM scores of Donald Trump



```
           Reference
Prediction -1  0  1
        -1  1  0  0
        0   6 14  3
        1   0  0  5

Overall Statistics

               Accuracy : 0.6897
                 95% CI : (0.4917, 0.8472)
    No Information Rate : 0.4828
    P-Value [Acc > NIR] : 0.01989

                  Kappa : 0.447
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: -1 Class: 0 Class: 1
Sensitivity            0.14286   1.0000   0.6250
Specificity            1.00000   0.4000   1.0000
Pos Pred Value         1.00000   0.6087   1.0000
Neg Pred Value         0.78571   1.0000   0.8750
Prevalence             0.24138   0.4828   0.2759
Detection Rate         0.03448   0.4828   0.1724
Detection Prevalence   0.03448   0.7931   0.1724
Balanced Accuracy      0.57143   0.7000   0.8125
>
```

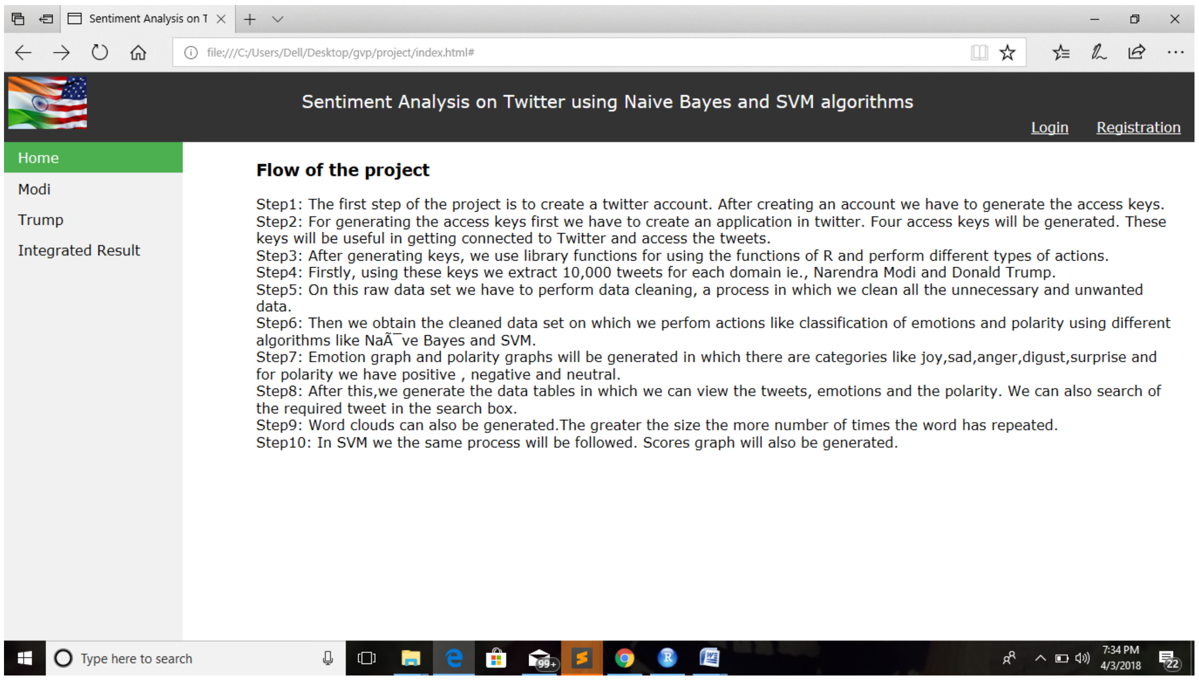
Fig 5.25 Confusion Matrix of Donald Trump



```
          Reference
Prediction  -1   0   1
        -1  65   0   0
        0   31 116  27
        1    0   0  60

Overall Statistics

               Accuracy : 0.806
                 95% CI : (0.7566, 0.8493)
    No Information Rate : 0.388
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6997
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: -1 Class: 0 Class: 1
Sensitivity             0.6771   1.0000   0.6897
Specificity             1.0000   0.6831   1.0000
Pos Pred Value          1.0000   0.6667   1.0000
Neg Pred Value          0.8675   1.0000   0.8870
Prevalence              0.3211   0.3880   0.2910
Detection Rate          0.2174   0.3880   0.2007
Detection Prevalence    0.2174   0.5819   0.2007
Balanced Accuracy       0.8385   0.8415   0.8448
```

Fig 5.26 Confusion Matrix of Narendra Modi

Front end Output Screens



Sentiment Analysis on Twitter using Naive Bayes and SVM algorithms

Login Registration

Home
Modi
Trump
Integrated Result

Flow of the project

Step1: The first step of the project is to create a twitter account. After creating an account we have to generate the access keys.
Step2: For generating the access keys first we have to create an application in twitter. Four access keys will be generated. These keys will be useful in getting connected to Twitter and access the tweets.
Step3: After generating keys, we use library functions for using the functions of R and perform different types of actions.
Step4: Firstly, using these keys we extract 10,000 tweets for each domain ie., Narendra Modi and Donald Trump.
Step5: On this raw data set we have to perform data cleaning, a process in which we clean all the unnecessary and unwanted data.
Step6: Then we obtain the cleaned data set on which we perfom actions like classification of emotions and polarity using different algorithms like NaÃ¯ve Bayes and SVM.
Step7: Emotion graph and polarity graphs will be generated in which there are categories like joy,sad,anger,digust,surprise and for polarity we have positive , negative and neutral.
Step8: After this,we generate the data tables in which we can view the tweets, emotions and the polarity. We can also search of the required tweet in the search box.
Step9: Word clouds can also be generated.The greater the size the more number of times the word has repeated.
Step10: In SVM we the same process will be followed. Scores graph will also be generated.

Fig 5.27 Front end home page

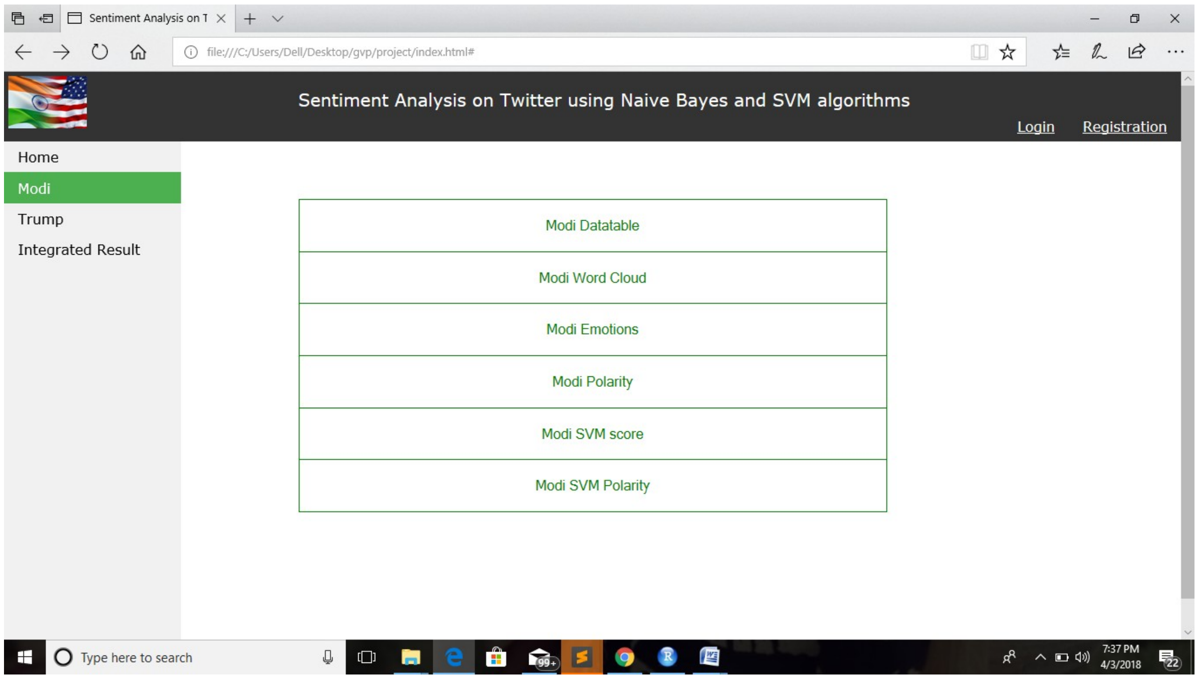
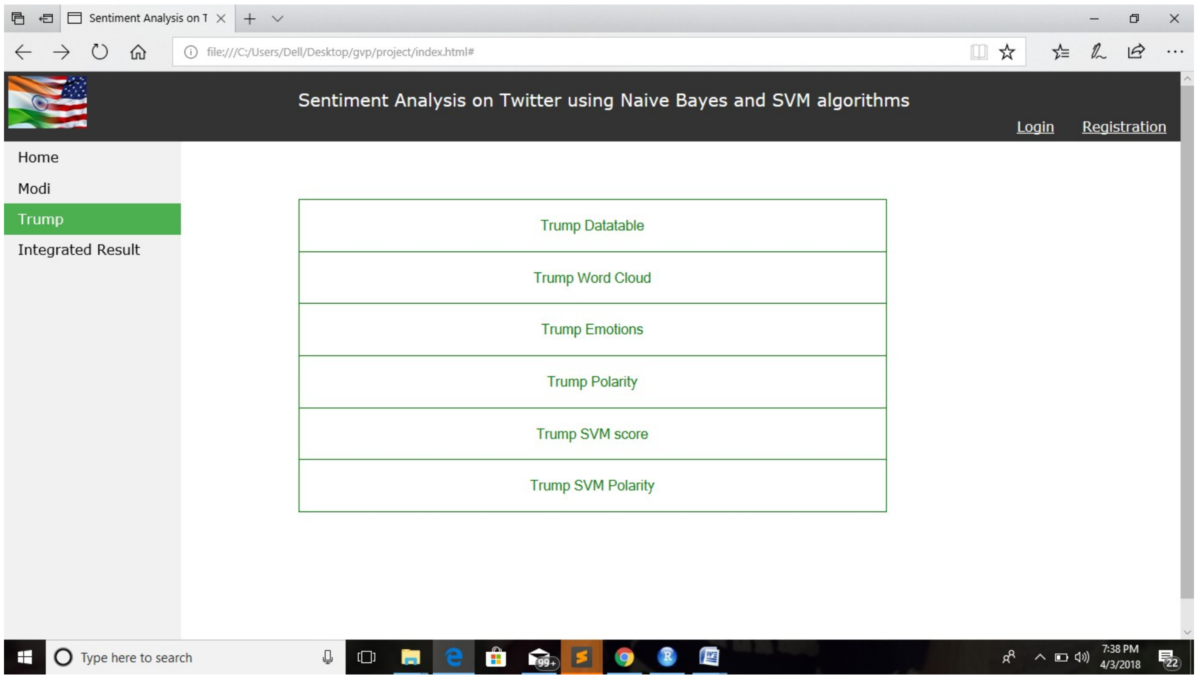


Fig 5.28 Front end Modi results page

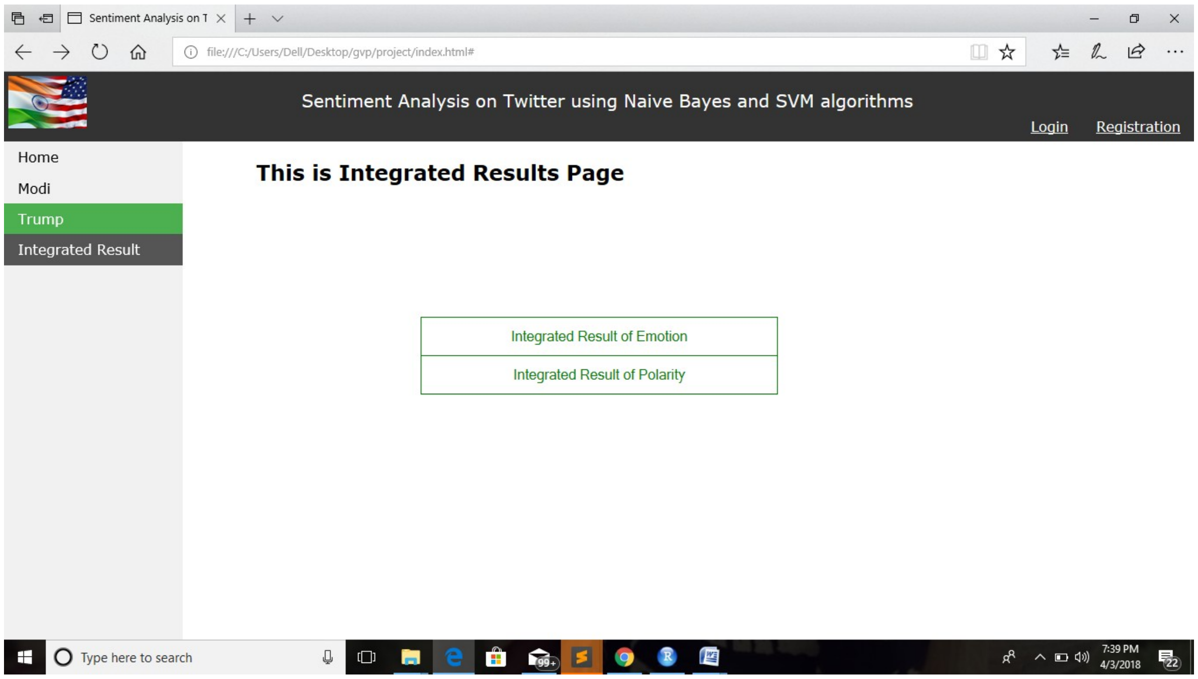

Sentiment Analysis on Twitter using Naive Bayes and SVM algorithms

Login Registration

Home
Modi
Trump
Integrated Result

Trump Datatable
Trump Word Cloud
Trump Emotions
Trump Polarity
Trump SVM score
Trump SVM Polarity

Fig 5.29 Front end Donald Trump results page


Sentiment Analysis on Twitter using Naive Bayes and SVM algorithms

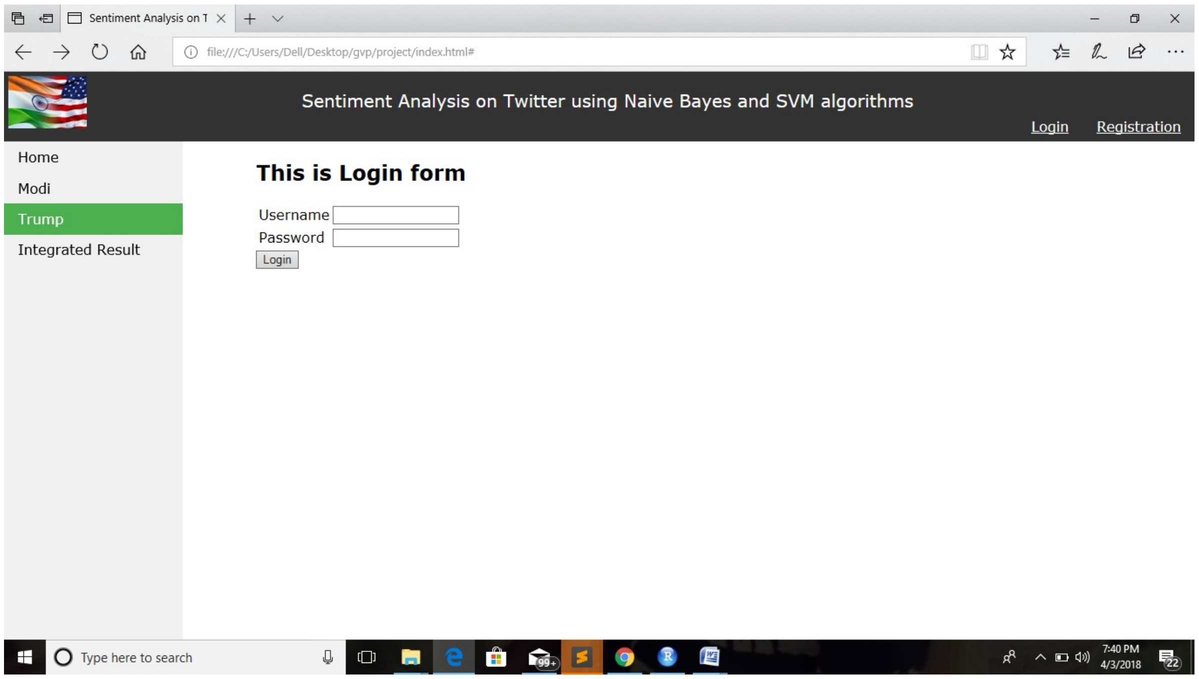
Login Registration

Home
Modi
Trump
Integrated Result

This is Integrated Results Page

Integrated Result of Emotion
Integrated Result of Polarity

Fig 5.30 Front end Integrated results page

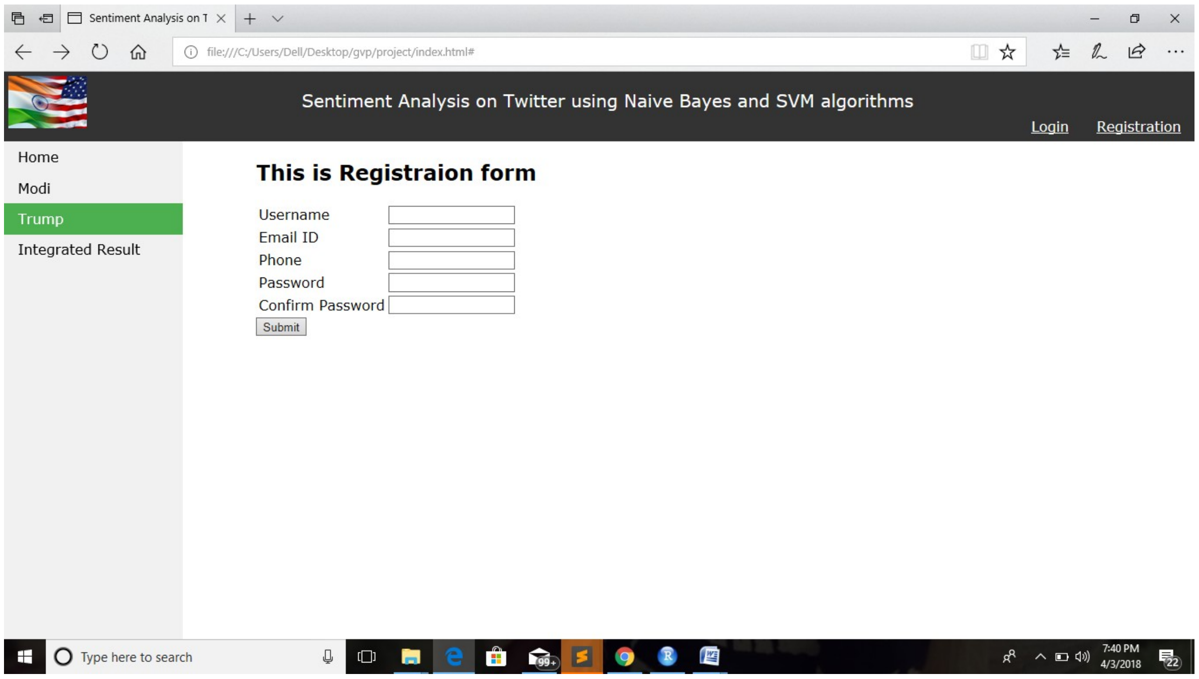


Sentiment Analysis on Twitter using Naive Bayes and SVM algorithms

This is Login form

Username	
Password	

Fig 5.31 Front end Login page



Sentiment Analysis on Twitter using Naive Bayes and SVM algorithms

This is Registraion form

Username	
Email ID	
Phone	
Password	
Confirm Password	

Fig 5.32 Front end Registration page

6. TESTING AND VALIDATION

6.1 INTRODUCTION

Testing is the major quantity control measure employed during software development. Its basic function is to detect errors in the software. During requirement analysis and design, the output is a document that is usually textual and non-executable. After the coding phase, computer programs are available that can be executed for testing purposes. This implies that testing not only has to uncover errors introduced during coding, but also uncovers requirement, design or coding errors in the program. Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software testing can also be stated as the process of validating and verifying that a software program/application/product:

1. meets the business and technical requirements that guided its design and development;
2. Works as expected; and
3. Can be implemented with the same characteristics.

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed. As such, the methodology of the test is governed by the software development methodology adopted.
Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers. In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

6.2 TESTING OBJECTIVES

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as-yet-undiscovered error.
- A successful test is that uncovers an as-yet- undiscovered error.

6.3 TEST LEVELS

The test strategy describes the test level to be performed. There are primarily three levels of testing: unit testing, integration testing, and system testing. In most software development organizations, the developers are responsible for unit testing. Individual testers or test teams are responsible for integration and system testing.

6.4 TYPES OF TESTING

6.4.1 UNIT TESTING

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements. Each module can be tested using the following two strategies:

6.4.2 BLACK BOX TESTING

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. In this testing only the output is checked for correctness. The logical flow of the data is not checked .This testing has been uses to find errors in the following categories:

a) Incorrect or missing functions
b) Interface errors
c) Errors in data structure or external database access
d) Performance errors
e)

6.4.3 WHITE BOX TESTING

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.

6.4.4 INTEGRATION TESTING

Integration testing ensures that software and subsystems work together as a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together.

6.5 VALIDATION

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

Test No.	**Test Case**	**Expected Output**	**Actual Output**	**Result**
1.	Invalid Log In Twitter*:* By providing invalid User name and Password	It displays error message saying enter a valid user name or password	It displayed error message saying username and password are incorrect	Success
2.	Valid Log In Test*:* By providing Valid User name and Password	Displays the available data sets	Displays the available data sets	Success
3.	Valid number of tweets in data table search box: By providing valid number(between 1-10000)	Displays the searched tweet in the data table asked by the user.	Displays the searched tweet in the data table asked by the user.	Success
4.	Invalid input in data table search box: By providing negative numbers, alphabets or numbers out of the given range(i.e.,>10000)	Display invalid input	Display invalid input	Failure

Table 6.5.1 UI Test Case

7. CONCLUSION

We can assume that the future of sentiment analysis will plug the existing gaps in being able to interpret meaning. There is an increased accuracy when compared to human processing. Sentiment analysis provides you with means to optimize your marketing strategy. People have already begun implementing a large-scale implementation of the sentiment analysis. This paves a way for many researchers to gain appropriate results. We need to think of sentiment analysis as "opinion mining," where the objective is to classify an opinion according to a polar spectrum. The extremes on the spectrum usually correspond to positive or negative feelings about something, such as a product, brand, or person." Sentiment is inherently subjective from person to person, and can even be outright irrational. Social media sentiment analysis can be an excellent source of information and can provide insights that can determine marketing strategy, improve campaign success, improve product messaging, and improve customer service.

8.REFERENCES

[1.] A.Pak and P. Paroubek. 'Twitter as a Corpus for Sentiment Analysis and Opinion Mining". In Proceedings of the Seventh Conference on International Language Resources and Evaluation, 2010, pp.1320-1326

[2.] R. Parikh and M. Movassate, "Sentiment Analysis of User- GeneratedTwitter Updates using Various Classi_cation Techniques",CS224N Final Report, 2009

[3.] Go, R. Bhayani, L.Huang. "Twitter Sentiment Classification Using Distant Supervision". Stanford University, Technical Paper,2009

[4.] http://ijarcs.info/index.php/Ijarcs/pages/view/pf

[5.] http://cran.rproject.org

[6.] https://www.youtube.com/watch?v=qWmMKmPVtgk

[7.] https://www.youtube.com/watch?v=_s4EKn_-uGo

[8.]https://colinpriest.com/2015/07/04/tutorial-using-r-and-twitter-to-analyse-consumer-sentiment/

[9.]https://sadanand-singh.github.io/posts/svmpython/

[10.]https://github.com/gastonstat/Mining_Twitter/blob/master/Rscripts/Sentiment_Analysis with_Starbucks.R

[11.]https://github.com/timjurka/sentiment/tree/master/sentiment/R

[12.]https://sites.google.com/site/miningtwitter/questions/sentiment/sentiment

[13.]https://rpubs.com/Leeladatascience/189957

[14.]http://www.springer.com/gp/book/9783319141411

[15.]https://arxiv.org/pdf/1601.06971

[16.]https://pdfs.semanticscholar.org/2bf4/7d8b55db138a0d25f5d8500dd44cba806faf.pd