SENTIMENT ANALYSIS USING LOGISTIC REGRESSION

A PROJECT REPORT

Submitted by

YADLA JASWANTH (111716104118)

RAVILLA MUNI SANDEEP KUMAR (111716104089)

RAVILLA MADHU SUDHAN (111716104088)

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RMK ENGINEERING COLLEGE, KAVARAIPETTAI

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2020

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that the project report "**Sentiment Analysis using Logistic Regression**" is the bonafide work of "**Y. Jaswanth (111716104118), R. Muni Sandeep Kumar (111716104089) and R. Madhu Sudhan (111716104088)**", who carried out the project under my supervision.

SIGNATURE

Dr. T. SETHUKARASI, M.E., M.S., Ph.D.,

PROFESSOR AND HEAD

Department of Computer Science and Engineering,
R.M.K. Engineering College,
R.S.M. Nagar,
Kavaraipettai-601206.

SIGNATURE

Mr. VIJAY KUMAR S,

SUPERVISOR,
ASSOCIATE PROFESSOR

Department of Computer Science and Engineering,
R.M.K. Engineering College,
R.S.M. Nagar,
Kavaraipettai-601206.

Submitted for the project viva voice held on……
at R.M.K. Engineering College, Kavaraipettai-601 206.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We earnestly portray our sincere gratitude and regard to our beloved **Chairman Shri. R. S. Muni Rathinam, our Vice Chairman, Mr. R. M. Kishore and our Director, Shri. R. Jyothi Naidu,** for the interest and affection shown towards us throughout the course.

We convey our sincere thanks to our **Principal, Dr. K. A. Mohamed Junaid,** for being the source of inspiration in this college.

We reveal our sincere thanks to our **Professor** and **Head of the Department,** Computer Science and Engineering, **Dr. T. Sethukarasi,** for her commendable support and encouragement for the completion of our project.

We would like to express our sincere gratitude for our **project coordinator Dr. Sandra Johnson, Professor, Mr. S. Vijayakumar, Associate Professor and Ms. Anusha Sanampudi, Assistant Professor** for their valuable suggestions towards the successful completion for this project in a global manner.

We convey our deep gratitude and we are very much indebted to our **versatile Project Guide Mr. Vijay Kumar S, Professor / Associate Professor /Assistant Professor** for his valuable suggestions and spontaneous guidance to complete our project.

We take this opportunity to extend our thanks to all faculties of the Department of Computer Science and Engineering, parents and friends for all that they meant to us during the crucial times of the completion of our project.

ABSTRACT

Sentiment Analysis is a sub-field of **Natural Language Processing (NLP)**. It is defined as a process of computationally identifying and categorizing opinions from a piece of text and is helpful in determining whether the writer's attitude towards a particular topic or product is positive, negative or neutral. It is also termed as Opinion Mining. The motive of our proposed work is to detect hate speech in tweets. We consider a tweet as hate speech if it has a negative sentiment. So, our objective is to categorize negative tweets from overall tweets.

Keywords: Natural language processing (NLP), Machine learning, Opinion mining, Hate speech

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. OVERVIEW

Data is categorized into 2 types.

1. Structured data
2. Unstructured data

Structured data is easily understandable and organised in an efficient manner, but it is not the case with Unstructured data. A large portion of data that is available on the internet is mostly in the unstructured format. Some examples of unstructured data are email messages, news articles, social media posts and audio files. Analysing such data can be done using Natural Language Processing (NLP). Sentiment Analysis is the most used field under NLP.

Sentiment Analysis is defined as a process of computationally identifying and categorizing opinions from the piece of text and determining whether the writer's attribute towards a particular topic or towards a product is positive, negative or neutral. Sentiment Analysis is also known as Opinion Mining.

Success of a company depends on the satisfaction of its users. Understanding how the customers respond to a product is very critical for further development of a business. So, if the customers like the product then it is considered a success, whereas if the users is not satisfied with the product then companies should start to analyse the flaws that it made by understanding the user's sentiment towards a product. Consider an example, where the mobile manufacture company produces a mobile, they wanted to know the feedback of the end users regarding the product. Manually reading the thousands of responses

about the product is not an efficient way, this is the place where Sentiment Analysis plays a key role in identifying the sentiment of the users regarding a product. Sentiment analysis is not only useful in the business domain but also can be useful in places where people often express their opinions on movies which can be helpful for filmmakers to know about the audience response [1]. In recent times, usage of social media and social networking sites are increasing exponentially. Comments, reviews and opinions of the people play an important role to determine whether a given population is satisfied with the product, services. It helps in predicting the sentiment of a wide variety of people on a particular event of interest like the review of a movie, their opinion on various topics roaming around the world. These data are essential for sentiment analysis. For, performing sentiment analysis[11], we use Twitter, a micro-blogging social networking website. It allows us to read short 140 characters length-based messages which are known as tweets. Twitter generates huge data that cannot be handled manually to extract some useful information and therefore, the ingredients of automatic classification are required to handle those data. By the use of Twitter, millions of people around the world are connected with their family, friends and colleagues through their computers or mobile phones. The Twitter interface allows the user to post short messages and that can be read by any other Twitter user. Twitter contains a variety of text posts and grows every day. We choose Twitter as the source for opinion mining simply because of its popularity and data mining. A number of research works are underway in examining the emotions to classify individual behaviours, responses or opinions [2].

There are 3 different ways in mining a sentiment, they are described as follows [3]:

1. **Document level sentiment analysis:** In this level, the whole document is categorized as positive, negative or neutral.

2. **Sentence level sentiment analysis:** In this level, every sentence is categorized as positive, negative or neutral. If there is a need to identify the particular sentence then this model will be useful in predicting sentiment.

3. **Aspect level sentiment analysis:** In this level, documents are categorized as positive, negative or neutral based on certain aspects. Aspect-based sentiment analysis is very helpful in cases such as for example consider a tweet "The camera is not that good in this mobile as it overexposes the images". In this situation, an aspect-based classifier can be able to detect that the user is mentioning a particular aspect (camera in this case) in a product that is defective.

Sentiment Analysis uses various NLP methods and algorithms. The main types of algorithms include [10]:

1. Rule Based Systems
2. Machine Learning Based Systems
3. Hybrid Approach

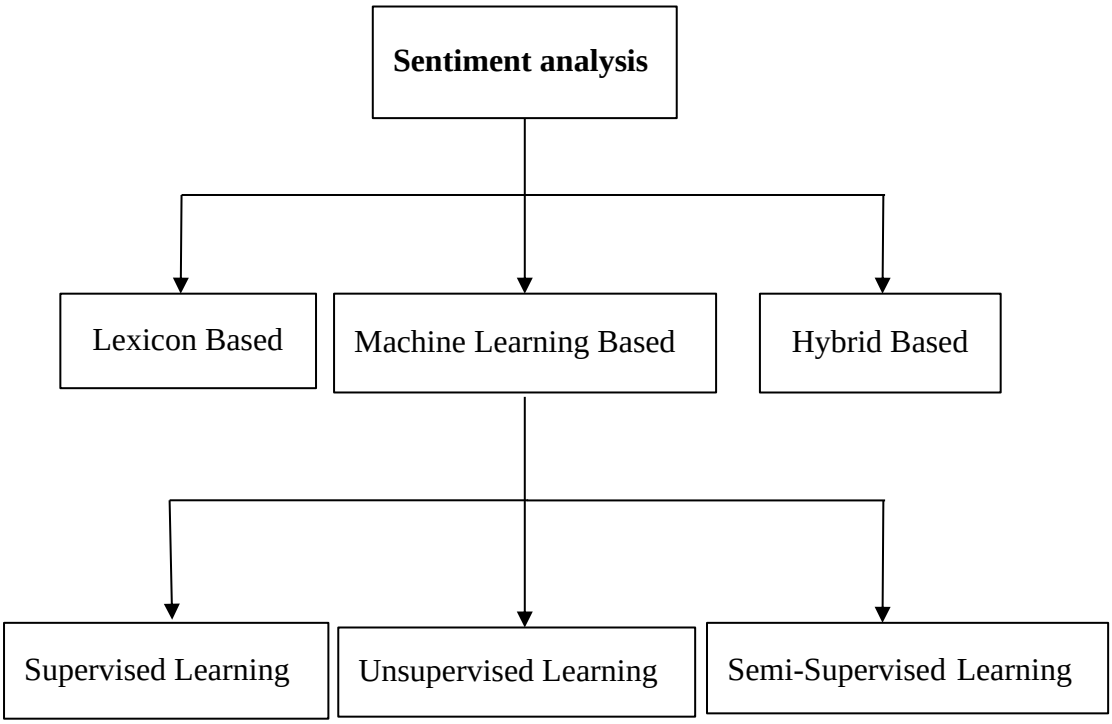

Fig 1.1 Various ways to perform Sentiment Analysis

1. **Rule Based Systems**

These systems depend on Manually crafted rules. This approach makes use of lexicons and some rules to know the polarity of a particular subject. Lexicons are the group of adverbs, adjectives and phrases which were scored by some individuals already. This scoring is said to be complex because different individuals give different sentiment scores to the same words based on their knowledge. Consider an example, one person would give a score of 0.6 to the word “good”, but some other person would give the same score for the word “great”. So, the scoring mechanism can be refined by calculating the mean of all the scores given for each word. This could be a time-consuming task and also the main drawback of this approach is to maintain the libraries consistently by adding the new words and phrases to it.

2.Machine learning Based Systems

These systems perform sentiment analysis based on the machine learning techniques to learn from the data. These systems do not require any manually crafted rules, but their base is machine learning techniques. By using them, we can process large volumes of data within no time. In this approach, initially we train the sentiment analysis model on giving the training data to it, so that it returns the polarity values on giving the new data. This approach yields better results in an efficient way compared to the rule-based approach.

In machine learning problems there are three types:

a. Supervised Learning
b. Unsupervised Learning
c. Reinforcement-Learning

A. Supervised Learning

Supervised Learning describes a class of problems that involves using a model to learn a mapping between input examples and the target variable.

Models are fit on training data composed of inputs and outputs and used to make predictions on test sets where only the inputs are provided and the outputs from the model are compared to the withheld target variables and used to estimate the skill of the model. In case of supervised machine learning approaches, to train the classification model training dataset is used which in turn will be helpful in classifying the test data [12].

There are two main types of supervised learning problems: they are classification that involves predicting a class label and regression that involves predicting a numerical value.

- **Classification:** Supervised learning problem that involves predicting a class label.

An example of a classification problem would be the MNIST handwritten digits dataset where the inputs are images of handwritten digits and the output is a class label for what digit the image represents (numbers 0 to 9).

- **Regression:** Supervised learning problem that involves predicting a numerical label.

An example of a regression problem would be the Boston house prices dataset where the inputs are variables that describe a neighbourhood and the output is a house price in dollars.

Both classification and regression problems may have one or more input variables and input variables may be any data type, such as numerical or categorical.

Algorithms are referred to as "supervised" because they learn by making predictions given examples of input data, and the models are supervised and corrected via an algorithm to better predict the expected target outputs in the training dataset.

B. Unsupervised Learning

Unsupervised Learning describes a class of problems that involves using a model to describe or extract relationships in data.

Compared to supervised learning, unsupervised learning operates upon only the input data without outputs or target variables. As such, unsupervised learning does not have a teacher correcting the model, as in the case of supervised learning.

There are many types of unsupervised learning, although there are two main problems that are often encountered by a practitioner: they are clustering that involves finding groups in the data and density estimation that involves summarizing the distribution of data.

- **Clustering:** Unsupervised learning problem that involves finding groups in data.

An example of a clustering algorithm is k-Means where *k* refers to the number of clusters to discover in the data.

- **Density Estimation**: Unsupervised learning problem that involves summarizing the distribution of data.

An example of a density estimation algorithm is Kernel Density Estimation that involves using small groups of closely related data samples to estimate the distribution for new points in the problem space.

Clustering and density estimation may be performed to learn about the patterns in the data.

Additional unsupervised methods may also be used, such as visualization that involves graphing or plotting data in different ways and projection methods that involves reducing the dimensionality of the data.

- **Visualization**: Unsupervised learning problem that involves creating plots of data.

An example of a visualization technique would be a scatter plot matrix that creates one scatter plot of each pair of variables in the dataset.

- **Projection**: Unsupervised learning problem that involves creating lower-dimensional representations of data.

An example of a projection method would be Principal Component Analysis that involves summarizing a dataset in terms of eigenvalues and eigenvectors, with linear dependencies removed.

C. Reinforcement Learning

Reinforcement Learning describes a class of problems where an agent operates in an environment and must *learn* to operate using feedback.

The use of an environment means that there is no fixed training dataset, rather a goal or set of goals that an agent is required to achieve, actions they may perform, and feedback about performance toward the goal.

It is similar to supervised learning in that the model has some response from which to learn, although the feedback may be delayed and statistically noisy, making it challenging for the agent or model to connect cause and effect.

Impressive recent results include the use of reinforcement in Google's AlphaGo in out-performing the world's top Go player.

Some popular examples of reinforcement learning algorithms include Q-learning, temporal-difference learning, and deep reinforcement learning.

3. **Hybrid Systems**

This system is a combination of both the rule based as well as Machine learning based approach.

Initially, Sentimental Analysis is performed only on the written paper documents, however, nowadays it is extensively used to identify the emotion of users on the internet from texts, tweets, blogs, news articles and movie reviews.

CHAPTER 2

LITERATURE SURVEY

Kamps et al. [3] utilised the lexical database WordNet [4] to find the sentiment associated with a word in various dimensions. They generated a distance metric on WordNet and found the semantic orientation of adjectives. WordNet databases have the words connected by synonyms. Their development in the distance measure can be applied only to the words in the components where similarity measures using the taxonomic hyponymy can be only applied to the noun and verb categories.

In 1970, Ekman et al. [5] did immense research on facial expressions. Their research work conveyed that universal facial expressions are enough to offer sufficient signs to detect emotions.

Akba et al. [6] made use of feature selection based on the information gain and chi-square metrics to elect the informative features after completing the process of stemming and lemmatization. The conducted experiments illustrated that associating feature selection metrics with SVM classifiers has improvements compared to previous studies.

Pak et al. [7] developed a twitter corpus by gathering tweets from Twitter API and interpreting those using emoticons accordingly. By taking that corpus, they built a sentiment classifier based on the multinomial Naive Bayes classifier which uses N-gram and POS-tags as features. In that method, there is a probability of occurring a mistake because tweet sentiments in the training set are labelled based on the polarity of emoticons. The training set is inefficient as it comprises only the tweets which are having emoticons.

A technique to get specifications such as battery, processor, camera for a specific product was developed in [8]. The main technical aspects of a product are found and classified. Depending on the polarity like positive, negative or neutral, scores were assigned for each and every specification. By combining all the scores of independent features, the overall rating of a product was identified.

For categorizing the feedback, an upgraded method from Support Vector Machine was proposed in [9]. Depending on the words associated with emotions, SentiWordNet assigned the sentiment scores. By changing these scores they developed a modified model.

Prabhsimran Singh, Ravindra Singh and Karanjeeet Singh Kalhon, [13] they have examined this government policy the demonetization from the ordinary person’s viewpoint with the use of the approach of sentiment analysis and using Twitter's data, Tweets are collected using certain hashtag (#demonetization). Analysis based on geo-location (State wise tweets are collected). The sentiment analysis API used from the meaning cloud and classified the states into six categories, they are happy, sad, very sad, very happy, neutral, and no data.

CHAPTER 3

EXISTING SYSTEM

Mining the user's opinion from social media is a difficult task, it can be done in numerous ways. Our Existing system has implemented sentiment analysis in the E-Healthcare domain using the R programming language. The tool they have used to analyze the sentiment of an user is by utilising a statistical tool, R programming. This sentiment analysis is based on the text data retrieval from streamed web and classifying people perspectives in eight different classifications of feelings (disgust, fear, anticipation, anger, sadness, trust, joy, surprise) and two unique sentiments such as positive and negative.

Some of the packages they have used for analysing user's sentiment is by using

1) twitterR : It is an R Package which is used for accessing the twitter data.

2) tm : A framework for text mining applications in R.

3) Syuzhet : Extracts sentiment and sentiment-derived plot arcs from text using a variety of sentiment dictionaries conveniently packaged for consumption by R users. This package allows users to choose a lexicon among four sentiment lexicons. The nrc lexicon concerns the polarity, i.e. reporting positive or negative words, it assigns a sentiment type, using the following 8 additional categories (disgust, fear, anticipation, anger, sadness, trust, joy, surprise). The occurrence in the text of a word appearing in one of the categories (say the disgust one) counts as 1 in the sentiment score for

that category. So, if a sentence contains 3 words listed in the list of words for disgust, the score for that sentence in the disgust category will be 3.

When using this approach, rather than receiving the algebraic score due to positive and negative words, each sentence gets a score for each sentiment category.

Drawbacks :

1. The major problem of this approach is, it does not properly consider negatives in the sentences. As a result for the sentence “he is not a good boy” and for the sentence “he is a good boy” the sentiment score will be the same for both the statements.
2. Manual updation of words in the corpus is a very difficult process.
3. Results are less accurate when compared to machine learning approaches.

CHAPTER 4

PROPOSED SYSTEM

Our proposed system is to detect the hate speech in the extracted tweets. We will categorize the tweet as hate speech if the tweet contains the negative sentiment. So, initially we have to identify the sentiment of the tweet as positive or negative, then we can classify all the negative tweets from the overall tweets. The reason for choosing hate speech detection is because cyber bullying has become a serious problem now-a-days in social media. So, Building a sentiment analysis model for detecting those texts can be helpful in using the internet. Initial step in our project is to gather tweets from twitter, and then we divide the dataset into 2 sets. The reason for dividing the dataset into 2 is because as we are using a machine learning approach, it means that the system makes decisions on its own, for a machine to have such power it should have some understanding. One is Training set and the other is the Testing set. As the name suggests, the Training set is used to train our machine learning model. It contains 3 columns namely id, label and tweets. Label column consists of binary values such as 0 and 1 where 0 denotes a non negative tweet and 1 denotes a negative tweet. We label them manually in the label column of the training set based on the Tweets column containing all the extracted tweets from twitter. Testing set contains only 2 columns namely id, tweets. The Tweets column contains the extracted tweets. Our objective is to name the label to the tweets present in the testing set. In our project, we use a Logistic regression algorithm to build a sentiment analysis model. This algorithm predicts the probability of occurrence of an event by fitting a data to the logit function.

4.1 Architecture diagram

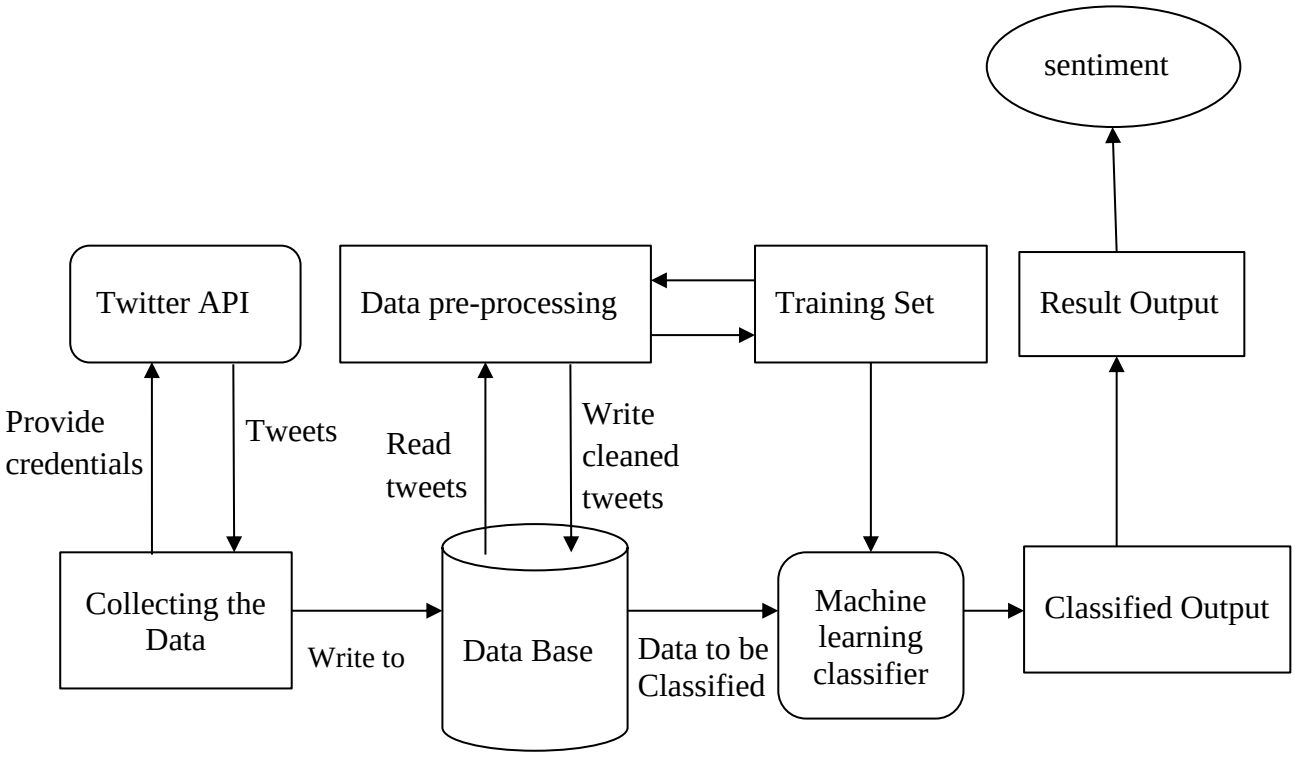

Fig 4.1 Data Flow Diagram

4.2 Required Python Packages

Before proceeding with the Methodology, we use certain python libraries which are useful for data analysis purposes.

1. **Numpy:**

   It is a powerful python library used for scientific computation and data manipulation in python. It efficiently implements multi dimensional arrays. It is an acronym for numerical python.

2. **Pandas:**

   Pandas stands for “Python Data Analysis Library”. It plays an important role in analysing the data and is one of the most preferred tools for data wrangling or data munging purpose. It takes the data (like a CSV or TSV file) and creates

a python object with rows and columns called a data frame that looks much similar to a table in statistical software. It is useful in opening local files such as CSV files.

3. **Matplotlib:**

   It is one of the most powerful tools for data visualisation in python. Matplotlib makes things easy and we can generate histograms, power spectra, bar charts, error charts etc by using it. It is most commonly used to translate complex data into digestible insights for a non-technical person. There are 2 main lines of code that are used, first is to define which chart we are going to use for example (bar chart, line chart etc) and second line is to display the graph.

4. **Seaborn:**

   It is a python library used for data visualisation which is based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

5. **nltk:**

   It is a python package used for Natural Language Processing (NLP). It contains text processing libraries like stemming, tokenization etc.

6. **Sklearn:**

   Scikit-learn provides a wide range of supervised and unsupervised learning algorithms via a constant interface in python. This library is used for modeling the data.

CHAPTER 5

METHODOLOGY

Sentiment classification can be done in the following ways:

1. Text pre-processing and cleaning
    a. Removing twitter handles
    b. Removing punctuations, numbers and special characters
    c. Removing short words
    d. Tokenization
    e. Stemming
2. Visualisation from tweets
3. Feature extraction from cleaned tweets
4. Building the model (training)

5.1 Text pre-processing and cleaning

Consider an example, there is an important document that you are searching in your room. Room is very clumsy with so many document papers. This makes your search process difficult. Suppose if all the documents are arranged in an organised manner in the room, then it will be very easy to find the document you wanted. Data cleaning is also a quite similar process. If data is arranged in a structured format then it will be easier to find the right information you needed in less time.

Text pre-processing is a key step in analysing large portions of data as it makes raw data ready for mining. If this step is not performed in an efficient way then there is a higher probability that we are dealing with scattered and inconsistent data. The objective of this step is to remove unrelated text from the tweets such as punctuations, user handles, Special characters.

a. Removing twitter handles

Initially we start the text pre-processing by removing the user names from the tweets as they are more in number because this is a way a twitter user is acknowledged in twitter. The twitter handles are hidden due to privacy issues. So, there will be no use of them for calculating sentiment from tweets. Hence, wherever the tweets containing @user symbol are removed from the tweets.

For example, the raw tweet looks like, “@user he always misguides people by propagating false news #Badpeople”.

After removal of twitter handle, the tweet look like “he always misguides people by propagating false news #Badpeople”.

b. Removing punctuations, numbers and special characters

Next step is to remove punctuations, numbers and special characters from the extracted tweets because these do not help in identifying sentiment of an user. So removing them from tweets is a better option similarly as we removed user handles from the tweets. Here we will replace everything excluding hashtags and characters with the spaces.

Consider an example where the raw tweet looks like “Apple has launched a new mobile “iphoneX” in an aggressive pricing category!!. It looks awesome…!!#Applefan”.

After removal of punctuation, special characters and numbers the tweet looks as “Apple has launched a new mobile iphoneX in an aggressive pricing category It looks awesome #Applefan”.

c. Removing short words

In every language, there are some words which are very common. While their use in the language is crucial, they do not convey any meaning, especially if taken out of context. This is the case of articles, conjunction and some adverbs which are commonly known as stop words. The next step in text pre-processing and cleaning is to remove short words such as ‘he’, ‘it’, ‘aw’, ‘oh’ etc as these words are not helpful in knowing an opinion of an user. We have to be a little careful in this step in choosing the length of words to remove. In our project, we decided to remove the words with words of length 3 or less than 3. So, words such as ‘hmm’, ‘oh’ are removed from the extracted tweets.

Consider an example, the raw tweet looks like “Rajesh is a good cricketer he plays cover drive greatly”.

After the removal of short words from the tweets then it looks like “Rajesh good cricketer plays cover drive greatly”. We can see from the example that the important words in the tweet have been preserved although the short words have been removed.

d. Tokenization

Tokenization is the process of splitting the text into tokens. Tokens are defined as sequences of characters which are combined together to form a semantic unit for pre-processing. In this step, we will tokenize the tweets in the dataset.

Consider an example where the raw tweet looks like “Rajesh hits slog shots brilliantly into stands”.

After tokenization the tweet looks as follows [Rajesh, hits, slog, shots, brilliantly, into, stands].

e. Stemming

Stemming is a process of removing suffixes from a word and bringing it to its root word. We use the popular Porter Stemmer algorithm for stemming. The reason for doing it is to cut down the word to its stem word because there will be no loss in meaning. For example, the words such as sing, singer, sings, singing all comes under the same root word sing.

Consider a tweet “I like watching movies on weekends”. After stemming, the tweet looks like “I like watch movie on weekend”. After all these steps the tokens are again retained back together.

5.2 Visualisation from tweets

Before we begin to train our machine-learning classifier there are certain steps that need to be addressed to gain insights about the data that we are dealing

with. This is an essential step because it gives answers to all the queries we have with the data. The questions that arise are

1. What are the most common words used in the extracted tweets?
2. What are the most common words that are used in negative and positive tweets in the dataset?
3. Are Hashtags really helpful in determining the sentiment?

Understanding the most common words in tweets using the Word Cloud

Word cloud is also known as Tag cloud. Tags are nothing but single words, the importance of each tag is displayed with font size or colour. Word Cloud is a means for depicting the keywords used in our classified data. It is a technique where most frequent words are displayed in large size and the least occurring words are displayed in smaller size. In simple words, word clouds are defined as a cluster of words depicted in different sizes. This is an ideal way to pull out the most pertinent parts of textual data, from blog posts to databases.

Fig 5.1 Word Cloud representation

Using word clouds we initially visualize all the keywords in the extracted tweets, later we again plot a word cloud for understanding the negative and

positive words in the tweets. By seeing these word clouds, we can understand the positive, negative words.

Understanding the impact of Hashtags in determining the sentiment of tweets

Hashtags on twitter have long been an important tool on twitter for helping users to organize their tweets. It is a great way to indicate that the content in tweets are relevant to a particular topic. Now, We should check whether these hashtags have any importance in identifying the sentiment of users in a sentiment analysis task. Searching process can be simplified by using these hashtags. Hashtags serving the relevant information in the tweets are considered as good hashtags. Hashtags are the simplest way to organize the endless amount of information posted on social media so that we see only the relevant topics. They are also helpful in reaching the tweets to many users who are interested in that particular content. They also help in distinguishing the two different sentiments.

For instance, let us consider a tweet:

"Iphone se has been launched at an affordable price range with great specifications and i am sooo happy #excited #applefanboy".

This tweet seems to be positive and the hashtag also gives the same feeling.
Now let us take another tweet,

"Physical abuse against women in film industry is getting increased day by day #metoo #psychopaths"

This tweet is indicating a negative sentiment and hashtag also has the same impact.

Initially we will extract all the hashtags from the tweets and store all the tweets in two separate lists, one for negative tweets and the other for positive tweets. After preparing the list of all the tweets for both the sentiments, we can plot the

top n hashtags. Firstly, we plot all the positive hashtags and then plot all the negative hashtags in a bar graph. In our dataset, hashtags serve the purpose of calculating the sentiment of an user for sentiment analysis tasks. The next step in our project is to extract features from the tweets.

5.3 Feature extraction from the cleaned tweets

In text pre-processing, documents are mostly in textual format categorizing becomes the most difficult process so to encode data that is ready to use for machine learning algorithms, the mapping of textual data to real valued vectors is called Feature extraction. There are several techniques used for feature extraction purpose:

1. Bag-of-words approach
2. TF-IDF approach
3. Word Embeddings
    a. Word2Vec
    b. Doc2Vec

1. BAG-OF-WORDS approach

It is a popular and simple method of extracting features on raw text. Bag-of-words is a representation of text that describes the occurrence of words in a document. This approach is also known as BOW-model. In simple terms, it's a collection of words to represent a sentence with word count and mostly disregarding the order in which they appear.

Bag-of-Words plays a major role in:

1. Natural Language Processing
2. Information retrieval from documents

3. Document classification

This approach involves two steps :

1. A vocabulary of Known words
2. A measure of presence of known words

The reason why this approach is called bag-of-words is because any information about the order or structure of words in the document is discarded. This model is only concerned with whether the known words occur in the document or not.

The intuition is that documents are similar if they have similar content. Further, that from the content alone we can learn something about the meaning of the document.

The bag-of-words can be as simple or complex as you like. The complexity comes both in deciding how to design the vocabulary of known words (or tokens) and how to score the presence of known words.

Consider a corpus (a collection of texts) called C of D documents {d1, d2, ….., dD} and N unique tokens extracted out of the corpus C. The N tokens (words) will form a list, and the size of the bag-of-words matrix M will be given by D X N. Each row in the matrix M contains the frequency of tokens in document D(i). We will take an example for explaining the bag-of-words approach in stepwise.

Step-1: Collect the data

Suppose we have 3 documents namely

D1: Kim is a very active person. He likes to play cricket.

D2: Raju is an introverted person.

D3: Smitha likes to play hockey on weekends.

Step-2: Design the vocabulary

After performing all the pre-processing steps such as removing stop words, tokenization, the list generated would consist of all the unique words in the corpus. These are the known words from the vocabulary

[Kim, very, active, person, likes, play, cricket, Raju, introverted, Smitha, hockey, weekends].

Step-3: Matrix representation

The objective is to turn each document of free text into a vector that we can use as input or output for a machine learning model.

Here D=3, N=12. As we know the matrix form in bag-of-words approach will be in the form of DXN. So, a matrix size of 3X12 will be represented as :

Table 5.1 Matrix Representation in Bag-of-Words approach

	Kim	very	active	person	likes	play	cricket	Raju	introverted	Smitha	hockey	weekends
D1	1	1	1	1	1	1	1	0	0	0	0	0
D2	0	0	0	1	0	0	0	1	1	0	0	0
D3	0	0	0	0	1	1	0	0	0	1	1	1

Managing the vocabulary

As vocabulary increases, the vector representation part becomes difficult.

In the above mentioned example, the length of the document vector is equal to the number of known words.

We can imagine that for a very large corpus, such as thousands of books, that the length of the vector might be thousands or millions of positions. Further, each document may contain very few of the known words in the vocabulary.

This results in a vector with lots of zero scores, called a sparse vector or sparse representation.

Sparse vectors require more memory and computational resources when modelling and the vast number of positions or dimensions can make the modelling process very challenging for traditional algorithms.

Limitations of Bag-of-Words

The bag-of-words model is very simple to understand and implement and offers a lot of flexibility for customization on your specific text data.

It has been used with great success on prediction problems like language modelling and documentation classification.

Nevertheless, it suffers from some shortcomings, such as:

- **Vocabulary**: The vocabulary requires careful design, most specifically in order to manage the size, which impacts the sparsity of the document representations.
- **Sparsity**: Sparse representations are harder to model both for computational reasons (space and time complexity) and also for information reasons, where the challenge is for the models to harness so little information in such a large representational space.

- **No Term Ordering**: Discarding word order ignores the context, and in turn meaning of words in the document (semantics). Context and meaning can offer a lot to the model, that if modelled could tell the difference between the same words differently arranged ("this is interesting" vs "is this interesting"), synonyms ("old bike" vs "used bike"), and much more.

2. TF-IDF approach

A problem with scoring word frequency is that highly frequent words start to dominate in the document (e.g. larger score), but may not contain as much "informational content" to the model as rarer but perhaps domain specific words.

One approach is to rescale the frequency of words by how often they appear in all documents, so that the scores for frequent words like "the" that are also frequent across all documents are penalized.

This approach to scoring is called Term Frequency – Inverse Document Frequency, or TF-IDF for short.

It is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus (large and structured set of texts).

- Term Frequency: is a scoring of the frequency of the word in the current document.
- Inverse Document Frequency: is a scoring of how rare the word is across documents.

Term frequency has the same meaning as explained in the Bag of words whereas the inverse document frequency is the inverse of the number of documents that a particular term appears or the inverse of the document frequency by compensating the rarity problem in BOW model. For example, let us consider two terms 'the' and 'wallet' from a document corpus. The document frequency of the term 'the' appeared to be frequent whereas the term 'wallet' is a rare term in the document. Thus, by taking the inverse of a document frequency the TF-IDF vectorizer gave the preference to the rarity of a word.

TF-IDF works by penalizing the common words by assigning them lower weights while giving importance to words which are rare in the entire corpus but appear in good numbers in few documents.
Let's have a look at the important terms related to TF-IDF:

- TF = (Number of times term t appears in a document)/(Number of terms in the document)
- IDF = log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in.
- TF-IDF = TF*IDF

Now let us take an example to explain this part clearly,

Consider we have 3 documents

D1: Krishna plays hockey.

D2: Ram plays cricket.

D3: Krishna cooks well.

$$W_{t,d} = \begin{cases} 1 + log_{10}\, tf_{t,d} & if\ tf_{t,d} > 0 \\ 0 & otherwise \end{cases}$$

In the above equation, when the term frequency is '1' the log value becomes '0'. In order to prevent weighted tf becoming '0' when term frequency is '1' a '1' is added to the log. For instance, if you want to calculate the weighted tf of 'good' term in document d1, you need to get tf of 'good' in document d1 which is 2 and apply in the equation.

The weighted tf values for above example is represented in the below table.

Table 5.2 Weighted TF values are represented in below matrix form

	Krishna	plays	hockey	Ram	cricket	cooks	well
D1	0.33	0.33	0.33	0	0	0	0
D2	0	0.33	0	0.33	0.33	0	0
D3	0.33	0	0	0	0	0.33	0.33

As mentioned earlier the log of inverse of the document frequency is considered as weighted IDF and the equation is depicted as log (N/df). Here N denotes the number of documents in the corpus and df denotes document frequency.

To calculate idf, we need to first calculate the document frequencies (df) of each term. ie. Number of documents containing a particular term. For instance 'hockey' contained in only 1 document D1, so the df of 'hockey' is 1. Then to calculate the weighted idf value we use this equation log(N/df). The weighted idf values for the above mentioned example are represented below in table.

Table 5.3 Weighted IDF values are represented as

	Krishna	plays	hockey	Ram	cricket	cooks	well
DF	2	2	1	1	1	1	1
IDF	0.176	0.176	0.477	0.477	0.477	0.477	0.477

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

Here i refers to the term t and j refers to document D. The calculated tf-idf values are depicted in the following table.

Table 5.4 TF-IDF values for the above taken example

	Krishna	plays	hockey	Ram	cricket	cooks	well
D1	0.058	0.058	0.157	0	0	0	0
D2	0	0.058	0	0.157	0.157	0	0
D3	0.058	0	0	0	0	0.157	0.157

Finally, we have the tf-idf vectors for the D1, D2 and D3 documents against the unique terms contained in the documents.

CHAPTER 6

ALGORITHM

Every Machine learning algorithm works fine under a set of conditions, if we make sure the algorithm fits the requirements then it ensures the superior performance. However, any algorithm cannot be used in any condition. For example, linear regression can not be used to deal with categorical data as it is used when the dependent variable is continuous and the nature of the regression line is linear. As our problem statement deals with detecting the hatred speech from the overall tweets which is a classification problem **Logistic Regression** will be more appropriate to use.

Logistic Regression:

Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables. To represent binary/categorical outcomes, we use dummy variables. You can also think of logistic regression as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as a dependent variable. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. We use the **Sigmoid function/curve** to predict the categorical value. The threshold value decides the outcome(win/lose).

Types of Logistic regression

- **Binary Logistic regression**
- **Multinomial Logistic regression**
- **Ordinal Logistic regression**

Binary Logistic regression: It has only two possible outcomes. Example- yes or no.

Multinomial Logistic regression: It has three or more nominal categories. Example-cat, dog, elephant.

Ordinal Logistic regression: It has three or more ordinal categories, ordinal meaning that the categories will be in an order. Example- user ratings (1-5).

In logistic regression, we are only concerned about the probability of outcome dependent variable (success or failure). To start with logistic regression, we will discuss the simple linear regression equation with dependent variables enclosed in a link function.

$$\mathbf{g(x)= \beta_0 + \beta(num)}$$

Here 'num' is an independent variable.

As described above, g() is the link function. This function is established using two things: Probability of Success(p) and Probability of Failure(1-p). p should meet following criteria:

1) It must always be positive (since p >= 0)
2) It must always be less than equals to 1 (since p <= 1)

Since probability must always be positive, we'll put the linear equation in exponential form. For any value of slope and dependent variable, the exponent of this equation will never be negative.

$$\mathbf{P = exponent\ (\beta_0 + \beta(num)) = e^{(\beta_0 + \beta(num))}}$$

To make the probability less than 1, we must divide p by a number greater than p. This can simply be done by

$$\mathbf{P} = \frac{e\ (\beta_0 + \ \beta(num))}{1 + e\ (\beta_0 + \beta(num))}$$

From the above we can write that

$$\mathbf{P = e^x / 1 + e^x}$$

Here **P** is the probability of success. This equation is called Logit Function.
If P is defined as the probability of success, then 1-p is the probability of a failure.

$$\mathbf{1\text{-}P = 1 - e^x / 1+e^x}$$

By the above 2 equation we can write as follows

$$\frac{P}{1-P} = e^x$$

Now we will take log on the both sides, then the equation turns out to be

$$\text{Log } (\frac{P}{1-P}) = X$$

After substituting the value of x in above equation we get,

$$\text{Log } (\frac{P}{1-P}) = \beta_0 + \beta(\text{num})$$

Here num is an independent variable. log(p/1-p) is the link function. Here $\frac{P}{1-P}$ is called an odd ratio. Whenever the logarithm value of odd ratio is found to be positive, then the probability of success is always more than 50 percent. A typical logistic model plot is shown below. You can see probability never goes below 0 and above 1.

6.1 Performance Metrics

Once after building a classification model, we need to evaluate how good the predictions made by our model are. There are some performance metrics which help us improve our models.

1) **Precision:** It is implied as the measure of the correctly identified positive cases from all the predicted positive cases. Thus, it is useful when the costs of False Positives are high.

$$\text{Precision} = \frac{True\ positive}{(True\ positive + False\ positive)}$$

2) **Recall**: It is the measure of the correctly identified positive cases from all the actual positive cases. It is important when the cost of False Negatives is high.

$$\text{Recall} = \frac{True\ positive}{(True\ positive + False\ negative)}$$

3) **Accuracy:** One of the more obvious metrics, it is the measure of all the correctly identified cases. It is most used when all the classes are equally important.

$$\text{Accuracy} = \frac{True\ positive + True\ negative}{(True\ positive + False\ positive + True\ negative + False\ negative)}$$

4) F1- Score: This is the harmonic mean of Precision and Recall and gives a better measure of the incorrectly classified cases than the Accuracy Metric.

$$\text{F1-Score} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

In our Project, we use F1- score as an evaluation metric for our machine learning classifier as Accuracy is used when the True Positives and True negatives are more important while F1-score is used when the False Negatives and False Positives are crucial.

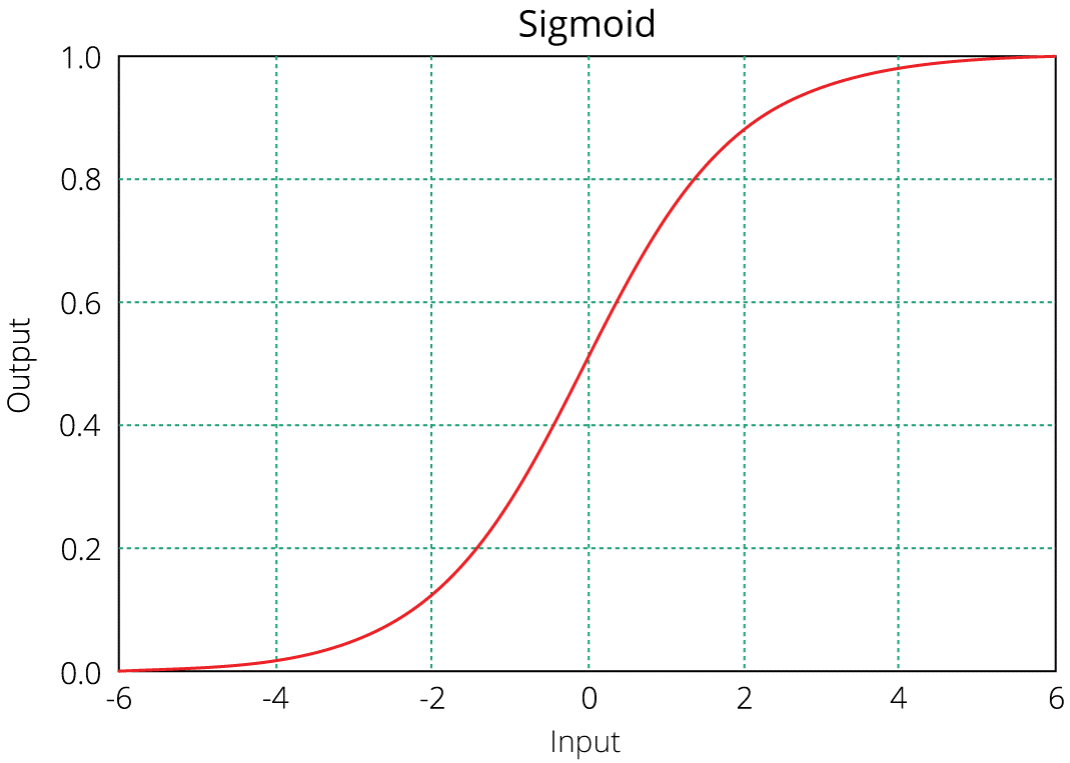

Fig 6.1 Sigmoid curve

CHAPTER 7

UML DIAGRAMS

7.1 Use Case Diagram

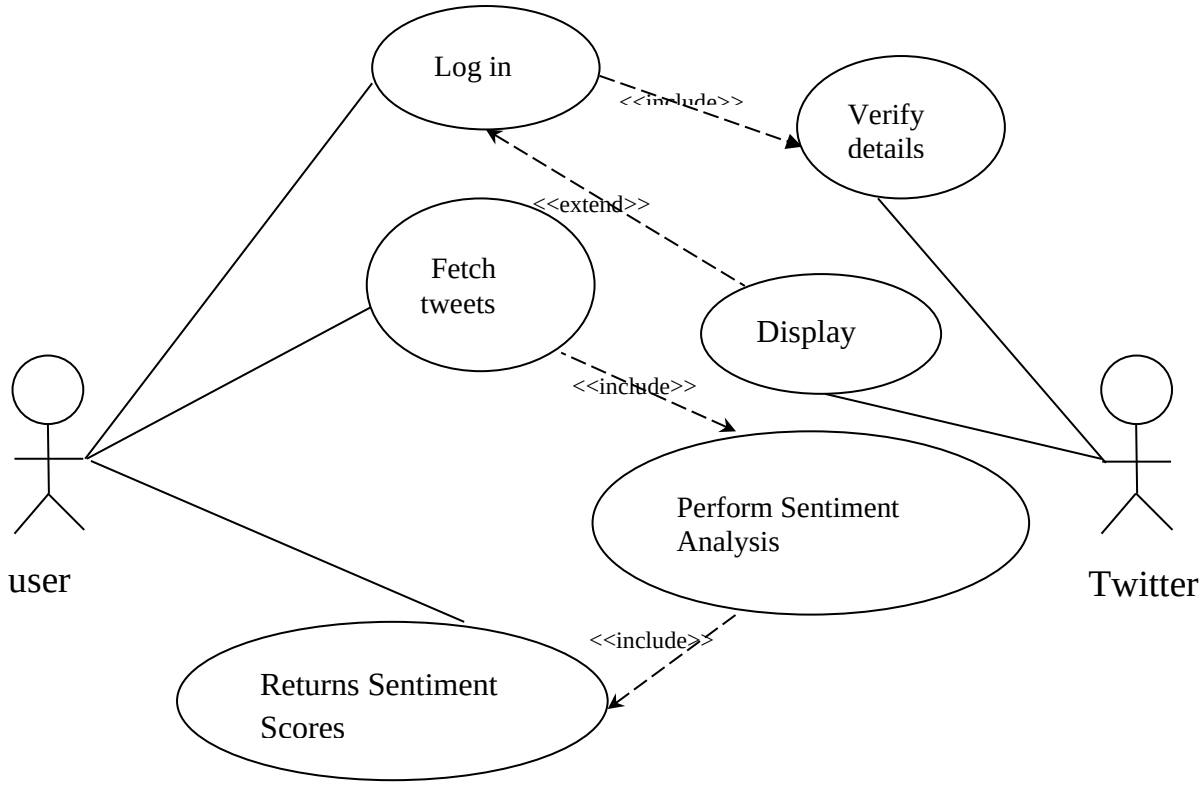

Fig 7.1 Use Case Diagram for Sentiment Analysis

7.2 Sequence Diagram

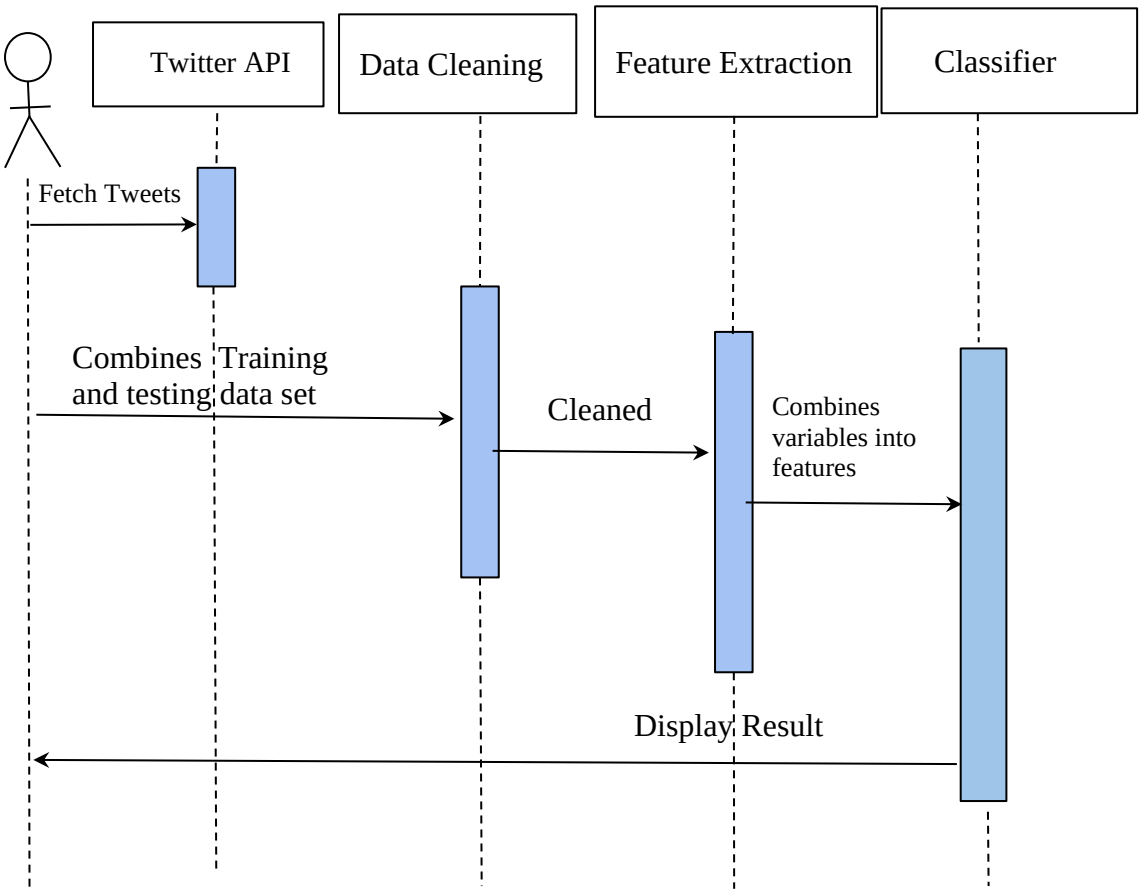

Fig 7.2 Sequence Diagram for Sentiment Analysis

CHAPTER 8

SYSTEM IMPLEMENTATION

8.1 Sample Code:

```
import re # for regular expressions
import pandas as pd
pd.set_option("display.max_colwidth", 200)
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import string
import nltk # for text manipulation
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)
%matplotlib inline

train_set= pd.read_csv('trainset copy final.csv')
test_set= pd.read_csv('tst2.csv')

test_set.head()
train_set.shape

#displaying some non-negative tweets
train_set[train_set['label'] == 0].head(10)

#displaying some negative tweets
train_set[train_set['label'] == 1].head(10)

#distribution of length of length of both train and test tweets
```

```
length_train_set = train_set['tweet'].str.len()
length_test_set = test_set['tweet'].str.len()
plt.hist(length_train_set, bins=20, label="train_Set_tweets")
plt.hist(length_test_set, bins=20, label="test_Set_tweets")
plt.legend()
plt.show()

#combining both train and test tweets
combine = train_set.append(test_set, ignore_index=True,sort=False)
combine.shape
def remove_pattern(input_txt, pattern):
  r = re.findall(pattern, input_txt)
  for i in r:
     input_txt = re.sub(i, '', input_txt)
  return input_txt

combine['cleaned_tweet']=np.vectorize(remove_pattern)(combine['tweet'],
"@[\w]*")
combine.head()

combine['cleaned_tweet']=combine['cleaned_tweet'].str.replace("[^a-zA-Z#]",  "
")
combine.head(10)

combine['cleaned_tweet'] = combine['cleaned_tweet'].apply(lambda x: ' '.join([w
for w in x.split() if len(w)>3]))
combine.head()

#Tokenization step
tokenized_tweet = combine['cleaned_tweet'].apply(lambda x: x.split())
tokenized_tweet.head()
```

```
#Stemming
from nltk.stem.porter import *
stemmer = PorterStemmer()

tokenized_tweet = tokenized_tweet.apply(lambda x: [stemmer.stem(i) for i in x])
for i in range(len(tokenized_tweet)):
  tokenized_tweet[i] = ' '.join(tokenized_tweet[i])
combine['cleaned_tweet'] = tokenized_tweet
combine.head()

#Visualisation of most frequented words using WordCloud
all_words = ' '.join([text for text in combine['cleaned_tweet']])
from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=21,
max_font_size=110).generate(all_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()

normal_words=''.join([text for text in combine['cleaned_tweet'][combine['label']
== 0]])
wordcloud=WordCloud(width=800,height=500,random_state=21,
max_font_size=110).generate(normal_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()

negative_words=' '.join([text for text in
combine['cleaned_tweet'][combine['label'] == 1]])
wordcloud = WordCloud(width=800, height=500,
```

```
random_state=21, max_font_size=110).generate(negative_words)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis('off')
plt.show()

#Defining a function to collect hashtags
def extract_hashtag(x):
  hashtags = []
  # Loop over the words in the tweet
  for i in x:
    ht = re.findall(r"#(\w+)", i)
    hashtags.append(ht)
  return hashtags
#extracting hashtags from non-negative tweets
HT_regular = extract_hashtag(combine['cleaned_tweet'][combine['label'] == 0])

# extracting hashtags from negative tweets
HT_negative = extract_hashtag(combine['cleaned_tweet'][combine['label'] ==
1])

# unnesting list
HT_regular = sum(HT_regular,[])
HT_negative = sum(HT_negative,[])
a = nltk.FreqDist(HT_regular)
d = pd.DataFrame({'Hashtag': list(a.keys()),'Count': list(a.values())})

# selecting top 20 most frequent hashtags
d = d.nlargest(columns="Count", n = 20)
plt.figure(figsize=(16,5))
ax = sns.barplot(data=d, x= "Hashtag", y = "Count")
ax.set(ylabel = 'Count')
```

```
plt.show()
b = nltk.FreqDist(HT_negative)
e = pd.DataFrame({'Hashtag': list(b.keys()), 'Count': list(b.values())})

# selecting top 20 most frequent hashtags
e = e.nlargest(columns="Count", n = 20)
plt.figure(figsize=(16,5))
ax = sns.barplot(data=e, x= "Hashtag", y = "Count")

#Extracting Features from cleaned tweets
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
import gensim
#Bag_of_Words Approach
bow_vectorizer=CountVectorizer(max_df=0.90,min_df=2, max_features=1000,
stop_words='english')
bow = bow_vectorizer.fit_transform(combine['cleaned_tweet'])
bow.shape
#TF_IDF Approach
tfidf_vectorizer= TfidfVectorizer(max_df=0.90, min_df=2, max_features=1000,
stop_words='english')
tfidf = tfidf_vectorizer.fit_transform(combine['cleaned_tweet'])
tfidf.shape
#Logistic Regression
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score

train_bow = bow[:31949,:]
test_bow = bow[31949:,:]
```

```
# splitting data into training and validation set
xtrain_bow, xvalid_bow, ytrain, yvalid = train_test_split(train_bow, train_set['label'], random_state=42,test_size=0.3)

lreg = LogisticRegression()
lreg.fit(xtrain_bow, ytrain) # training the model
prediction = lreg.predict_proba(xvalid_bow) # predicting on the validation set
prediction_int = prediction[:,1] >= 0.3 # if prediction is greater than or equal to 0.3 then 1 else 0
prediction_int = prediction_int.astype(np.int)

f1_score(yvalid, prediction_int) # calculating f1 score

test_pred = lreg.predict_proba(test_bow)
test_pred_int = test_pred[:,1] >= 0.3
test_pred_int = test_pred_int.astype(np.int)
test_set['label'] = test_pred_int
submission = test_set[['id','label','tweet']]
submission.to_csv('sub_lreg_bow1.csv', index=False)
submission.shape

train_tfidf = tfidf[:31949, :]
test_tfidf = tfidf[31949:, :]


xtrain_tfidf = train_tfidf[ytrain.index]
xvalid_tfidf = train_tfidf[yvalid.index]

lreg=LogisticRegression()
lreg.fit(xtrain_tfidf, ytrain)

prediction = lreg.predict_proba(xvalid_tfidf)
```

```
prediction_int = prediction[:,1] >= 0.3
prediction_int = prediction_int.astype(np.int)

f1_score(yvalid, prediction_int)
```

8.2 Sample Screenshots

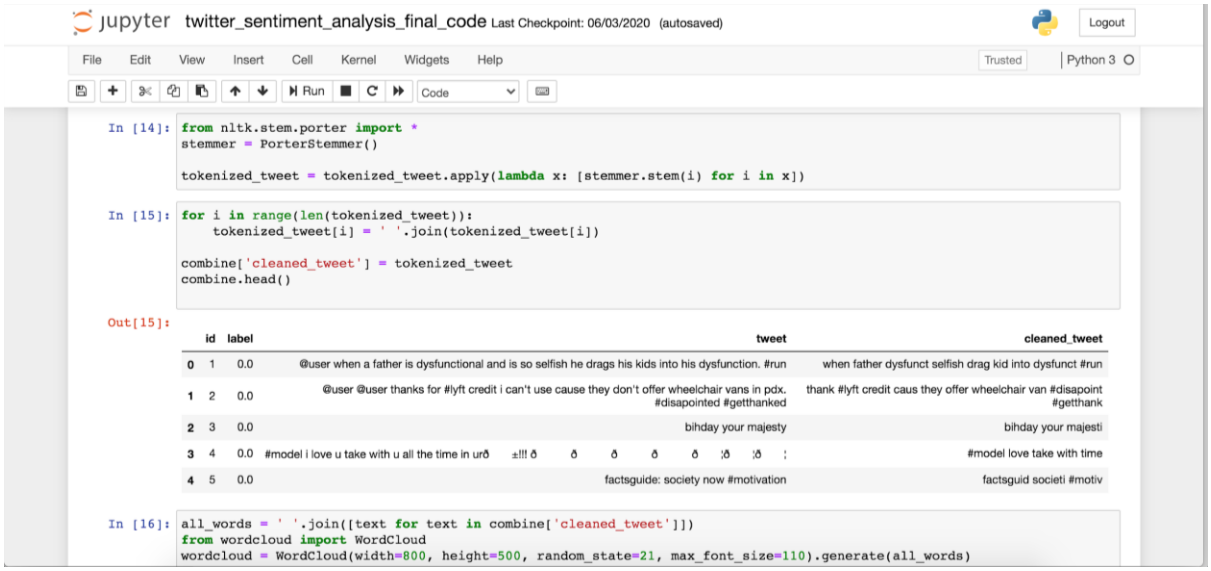

Fig 8.1 Top 5 tweets after Data cleaning

After performing all the data cleaning steps such as removing stopwords, removing punctuations, removing twitter handles, tokenizing data, after performing stemming the data looks like in the cleaned_tweet column.

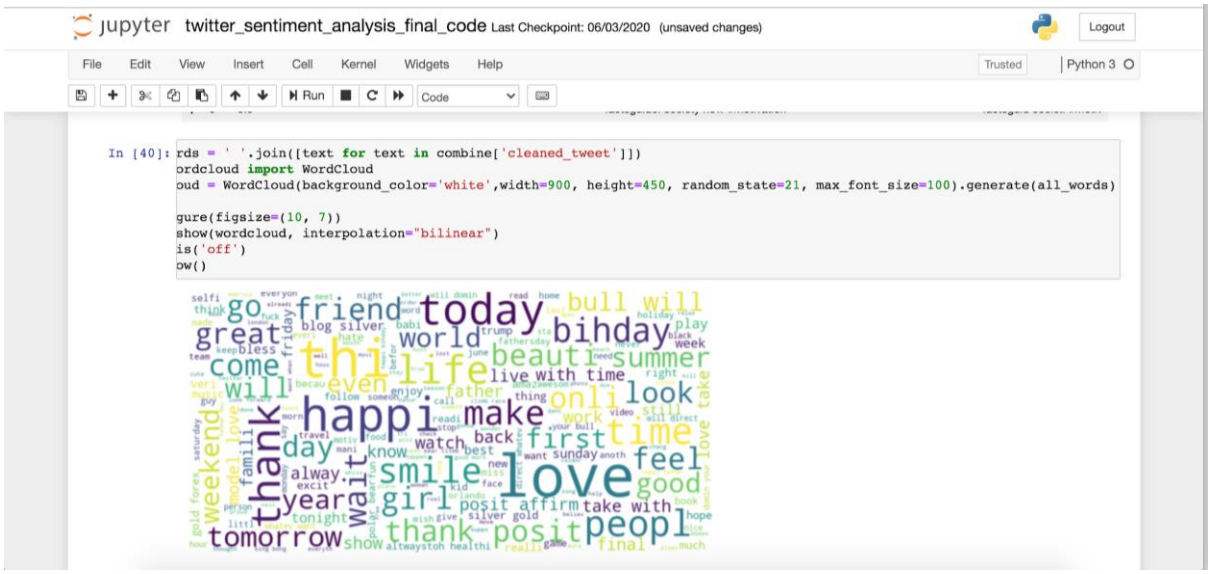

Fig 8.2 Representing all the frequently used words in dataset using Word Cloud

Word Cloud represents all the most frequently used words in the tweets after cleaning them. The above fig shows the word cloud of all the most frequent words in the dataset.

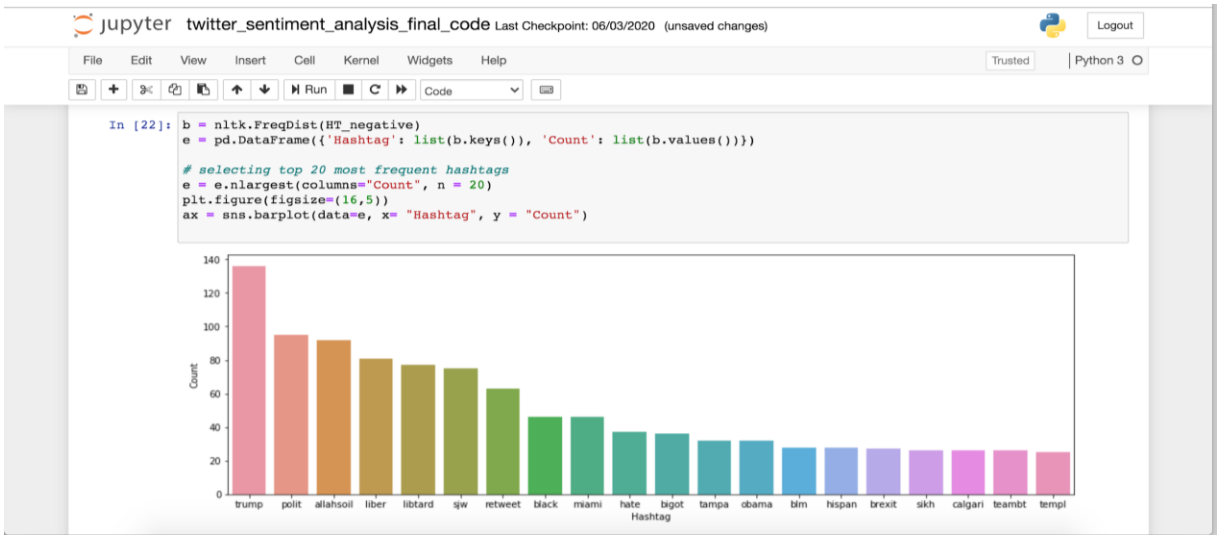

Fig 8.3 Top 20 most Frequent Negative Hashtags using Frequency Distribution

The above frequency distribution fig represents the most used negative emotion hashtags in the cleaned_tweet where x-axis represents the hashtag and y-axis represents the number of times the hashtag has appeared in the cleaned_tweet.

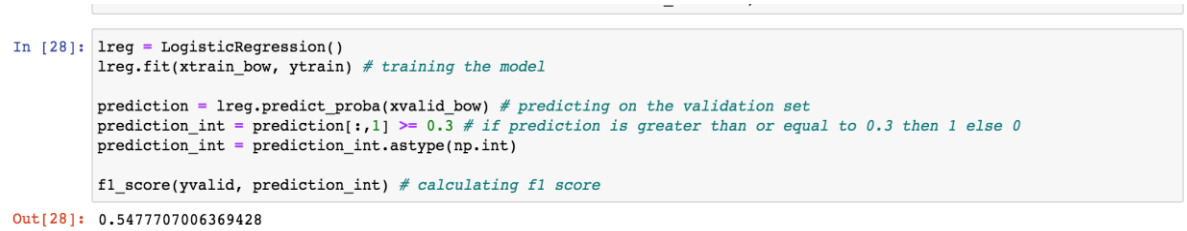

Fig 8.4 F1-Score of a classifier using **BOW** Approach

The above fig represents the f1-score of our sentiment analysis classifier using the bag of words feature extraction approach on the validation set.

```
[31]: lreg=LogisticRegression()
      lreg.fit(xtrain_tfidf, ytrain)

      prediction = lreg.predict_proba(xvalid_tfidf)
      prediction_int = prediction[:,1] >= 0.3
      prediction_int = prediction_int.astype(np.int)

      f1_score(yvalid, prediction_int)

t[31]: 0.5586776859504133
```

Fig 8.5 F1-Score of a Classifier using **TF-IDF** Approach

The above fig represents the f1-score of our sentiment analysis classifier using the tf-idf feature extraction approach on the validation set.

CHAPTER 9

CONCLUSION AND FUTURE WORK

As defined, Sentiment Analysis is used for identifying user's attitude toward a particular product is either positive or negative. It is deployed in several fields such as market research, product feedback, customer service etc. We used sentiment analysis for detecting the hate speech in the tweets using a machine learning algorithm called Logistic Regression.

Our Future work includes to deploy sentiment analysis in other platforms such as in YouTube and Restaurants so that it can be helpful for the owners to understand what the user is feeling and can rectify their mistakes. There are also some challenges that we faced during this project like detecting sarcasm, emoji's in tweets. This can be overcome by using more sophisticated machine learning algorithms and using various feature extraction methods.

CHAPTER 10

REFERENCES

[1] Svetlana Kiritchenko, Xiaodan Zhu, Saif M. Mohammad Sentiment Analysis of Short Informal Texts. Journal of Artificial Intelligence Research 50 (2014) 723-762.

[2] Walaa Medhat a,* Ahmed Hassan b, Hoda Korashy b Sentiment analysis algorithms and applications:A survey.Ain Shams Engineering Journal. Page no-1094.

[3] J. Kamps, M. Marx, R. J. Mokken, and M. De Rijke, "Using wordnet to measure semantic orientations of adjectives," 2004.

[4] C. Fellbaum, "Wordnet: An electronic lexical database (language, speech, and communication)," 1998.

[5] P. Ekman , "Universal facial expressions of emotion," Culture and Personality: Contemporary Readings/Chicago, pp. 151–158, 1974.

[6] F. Akba, A. Uçan, E. A. Sezer, and H. Sever, "Assessment of feature selection metrics for sentiment analysis: Turkish movie reviews," in 8th European Conference on Data Mining, 2014, vol. 191, pp. 180-184

[7] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining," in Proceedings of LREC, vol. 2010, 2010.

[8] M Vamsee Krishna Kiran, R. E. Vinodhini, R. Archanaa, K. VimalKumar, "User specific product recommendation and rating system by performing sentiment analysis on product reviews", 207 4th International Conference on Advanced Computing and Communication Systems (ICACCS), pp.1-5, 2017.

[9] Jasmine Bhaskar, Sruthi k, Prema Nedungadi, "Enhanced sentiment analysis of informal textual communication in social media by considering objective word and intensifiers", International Conference on Recent Advances and Innovations in Engineering.

[10] Z. Madhoushi, A. R. Hamdan and S. Zainudin, "Sentiment analysis techniques in recent works," 2015 Science and Information Conference (SAI), London, 2015, pp. 288-291.

[11] Federico Neri, Carlo Aliprandi, Federico Capeci, Montserrat Cuadros, Tomas, "Sentiment Analysis on Social Media", 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM).

[12] Jagdale, Rajkumar S., Vishal S. Shirsat, and Sachin N. Deshmukh. "Sentiment Analysis of Events from Twitter Using Open Source Tool." (2016).

[13] Singh, Prabhsimran, Ravinder Singh Sawhney, and Karanjeet Singh Kahlon. "Sentiment analysis of demonetization of 500 & 1000 rupee banknotes by Indian government." ICT Express (2017).