**Twitter Sentiment Analysis**

Thesis Submitted in fulfillment of the Requirement for the degree

Of

**Bachelor of Technology (B.Tech)**

**in**

**Department of Computer Science and Engineering**

**By  
Anik Dutta**

**(Roll No: 500117011125 Reg. No: 171430110480)**

**Subhadeep Das**

**(Roll No: 500117011107 Reg. No: 171430110107)  
 Susmita Goswami**

**(Roll No: 500117021112 Reg. No: 171430110114)  
 Purbasha Hatui**

**(Roll No: 500117021068 Reg. No: 171430110069)  
 Udayaditya Sasmal**

**(Roll No: 500117011119 Reg. No: 171430110122)**

**C:\Users\SONY\Desktop\logo.png**

**Guru Nanak Institute of Technology, Kolkata-700110**

**ACKNOWLEDGEMENT**

This project required an immense guidance, suggestions and monitoring from Mrs. Ipsita Saha, HOD (CSE) for successfully completing the project. We thank her for the help and guidance given by her that will carry us a long way in the journey of life on which we are about to embark.

Also we are grateful to our respected mentors Mr. Nirupam Saha and Mrs. Sayani Chandra for encouraging and supporting us throughout the project work.

|  |
| --- |
| A picture containing drawing, group, sign, graffiti  Description automatically generated  ---------------------------------------------  (Anik Dutta) |
| -----------------------------------------------  (Subhadeep Das) |
| ------------------------------------------------  (Susmita Goswami) |
| -------------------------------------------------  (Purbasha Hatui) |
| -------------------------------------------------  (Udayaditya Sasmal) |

C:\Users\SONY\Desktop\logo.png

GURU NANAK INSTITUTE OF TECHNOLOGY, KOLKATA

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

Certified that this project report **“Twitter Sentiment Analysis”** is the bonafide work of **“ Anik Dutta, Subhadeep Das, Susmita Goswami, Purbasha Hatui, Udayaditya Sasmal ”** who carried out the project work under my supervision.

**Head of the Department**

**----------------------------------**

**Mrs. Ipsita Saha**

Head of the Department

Department of Computer Science & Engineering,

GNIT, Kolkata

TABLE OF CONTENTS

**CHAPTER NO. TITLE PAGE NO.**

ABSTRACT i

LIST OF Abbreviations ii

LIST OF Nomenclature iii

**1. INTRODUCTION 1**

**2. TWITTER SENTIMENT ANALYSIS**

2.1 Literature Review 2

2.2 Problem Statement 3

2.3 ProposedSystem 4

2.4 Exploratory Data Analysis 5

2.4.1 Data Inspection 6

2.4.2 Data Cleaning & Processing 8

2.4.3 Data Visualization 11

2.5 Implementation 15

2.6 Experimental Results 19

**3. Conclusion and Future Study 22**

**4. References 23**

**ABSTRACT**

In this era of information people are just a click away from getting a huge chunk of information. Social media has opened a whole new realm for people around the world. People are getting information about different things and with this information comes people’s opinion. People may have positive and negative outlook regarding a topic. Sometimes this results into bullying and passing on hate comments about someone or something. To detect the sentiment associated with a particular comment, sentiment analysis is used.

**LIST OF Abbreviations**

NLP: Natural Language Processing

NLTK: Natural Language Toolkit

API: Application Package Interface

HTML: Hyper Text Markup Language

CSS: Cascading Style Sheet

**LIST OF Nomenclature**

Token: Token is an individual term or word.

Tokenization: It is the process of splitting a string of text into tokens.

Stemming: Stemming is the process of reducing a word to its word stem or root word.

Wordcloud: Wordcloud is a visualization in which the most frequent words appear in large size and the less frequent words appear in smaller sizes.

Precision : Precision is the intuitive ability of the classifier not to label as positive a sample that is negative.

Recall: Recall is the intuitive ability of the classifier to find all the positive samples.

F1 score: It is a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0.

True Positive Rate (TPR): TPR is the rate of actually positive values that are predicted as positive values.

False Positive Rate (FPR): FPR is the rate of actually negative values that are predicted as positive values.

False Negative Rate (FNR): FNR is the rate of actually positive values that are predicted as negative values.

True Negative Rate (TNR): TNR is the rate of actually negative values that are predicted as negative values.

Flask: Flask is a lightweight web frame of Python which provides the user with libraries, modules and tools to help build Web-Applications.

**Introduction**

Twitter is a popular microblogging service in which users post status messages called “tweets”. The millions of statuses appear on social networking every day. In most cases, its users enter their messages with much fewer characters than the limit established. Twitter represents one of the largest and most dynamic datasets. The datasets contains user generated content approximately 200 million users post 400 million tweets per day. Tweets can express opinions on various topics that can help to direct marketing campaigns so as to share consumers’ opinions concerning brands and products, outbreaks of bullying, events that generate insecurity, polarity prediction in political and sports discussions, and acceptance or rejection of politicians. In such application domains, one deals with large text corpora and most often “formal language”. At least two specific issues should be addressed in any type of computer-based tweet analysis: firstly, the frequency of misspellings and slang in tweets is much higher than that in other domains. Secondly, Twitter users post messages on a variety of topics unlike blogs, news, and other sites, which are tailored to specific topics. Big challenges can be faced in tweet sentiment analysis: a) neutral tweets are way more common than positive and negative ones. This is different from other sentiment analysis domains like product reviews which tend to be predominantly positive or negative, b) there are linguistic representational challenges, like those that arise from feature engineering issues and c) tweets are very short and often show limited sentiment cues.

**Literature Review**

There were many studies in sentiment analysis but almost those focused on a part of texts. A tweet is only limited to 140 characters. Bing Liu (2010), Tang and colleagues (2009) expressed an overview in sentiment analysis in which analyzed the strong points and the weak points of sentiment analysis and they gave many research ways of sentiment analysis. Pang and Lee (2004, 2008) compared many classifiers on movie reviews and gave a vision of insight and comprehension in sentiment analysis and opinion mining. Authors also used star rating as a feature for classification. Go et al (2009) studied on Bigram and POS. They removed emoticons out from their training data for classification and compared with Naive Bayes, MaxEnt and Support Vector Machine (SVM). They evaluated that SVM outperforms others. Barbosa and Feng (2010) pointed that N-gram is slow, so they researched on Microblogging features. Agarwal et al (2011) approached Microblogging, POS and Lexicon features, also they built tree kernel to classify tweets and applied on POS and N-Gram. Akshi Kumar and Teeja Mary Sebastian (2012) approached a dictionary method for analyzing the sentiment polarity of tweets. On the other hand, Stanford University (2013) performed a twitter sentiment classifier based on Maximum Entropy and built a Recursive Deep Model with a Sentiment Tree Bank.

**Problem Statement**

From a given tweet, classify whether the tweet is of positive or negative sentiment. The tweet which is conveying both positive and negative sentiment, whichever is stronger that should be taken.

**Proposed System**

Sentiment analysis (also known as opinion mining) is one of the many applications of NLP. It is the technique for extracting subjective information from text or speech, such as opinions or attitudes. In other words, it involves classifying a piece of text as positive, negative or neutral. We will use a Machine Learning model with NLP to perform the sentiment analysis. By using the model we will process the tweets and implement an algorithm for automatic classification of those tweets into positive or negative.

**Exploratory Data Analysis**

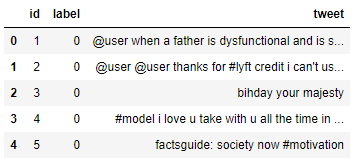
We are using a dataset from Kaggle.

There are 3 columns in the dataset.

1. id

2. label (where 1= Racist/sexist and 0= Non racist/sexist)

3. tweet



Text is a highly unstructured form of data, various types of noise are present in it and the data is not readily analyzable without any pre-processing. The entire process of cleaning and standardization of text, making it noise-free and ready for analysis is known as text preprocessing.

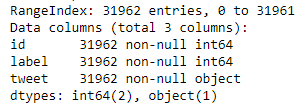
EDA can be divided it into 3 parts:

* Data Inspection
* Data Cleaning
* Data Visualization

**i)Data Inspection**

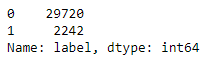
The dimension of the dataset is (31962, 3).

Then we check if there is any missing value in the dataset.



We find that there is no missing value.

Now let’s check the label-distribution in the dataset.



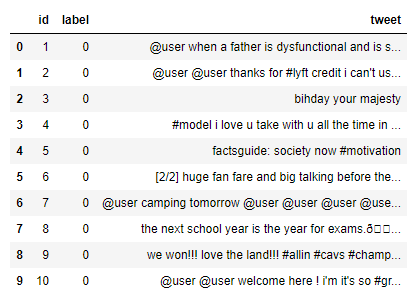
In the dataset, we have 2,242 tweets labeled as racist or sexist and 29,720 tweets labeled as non racist/sexist. We can clearly see that there is class imbalance in the dataset.

So we will remove the class imbalance. After removing the class imbalance we can see



Also the dimension of the dataset becomes (59440, 3).

Now let’s see 10 non racist/sexist tweets in the dataset.



Then we will check out 10 racist/sexist tweets.



There are quite a many words and characters which are not really required. So, we will try to keep only those words which are important and add value.

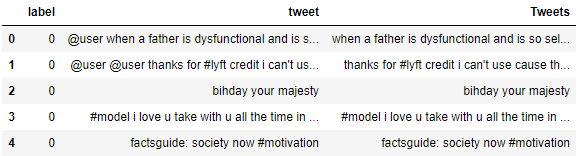
**ii) Data Cleaning & Processing**

After the data inspection, we have to perform data cleaning. We need to clean raw text data to get rid of the unwanted words and characters which helps in obtaining better features. If we skip this step then the data will be noisy and inconsistent. So we have to remove less relevant things such as punctuation, special characters, numbers, and terms which don’t carry much weightage in context to the text. Also there is a column named “id” that is not necessary for the sentiment analysis.

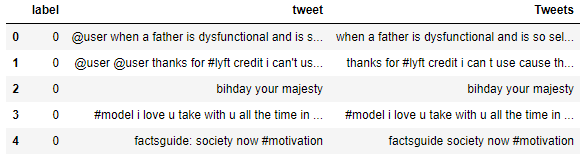
So first we will remove the column “id”.

We will be following the steps below to clean the raw tweets in our data.

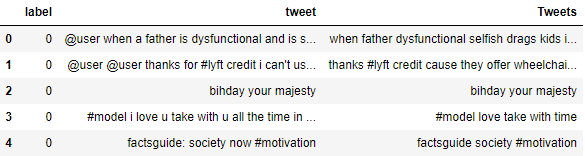
1. We will remove the twitter handles as these twitter handles hardly give any information about the nature of the tweet.



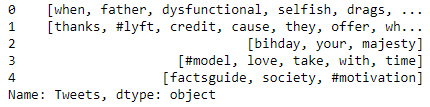
1. We will also get rid of the punctuations, numbers and even special characters since they wouldn’t help in differentiating different types of tweets.



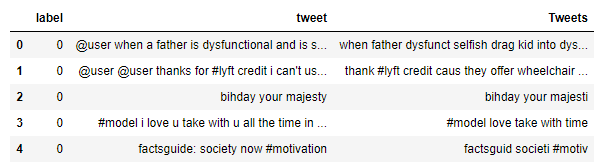
1. Most of the smaller words like ‘and’, ‘his’, ‘all’,’u’ etc. do not add much value in the sentiment analysis. So, we will try to remove them from our data.



1. Now we will have to tokenize the tweets. Tokens are individual terms or words and tokenization is the process of splitting a string of text into tokens.



1. Then we will normalize the text data like reducing terms like loves, loving, and lovable to their root word ‘love’. Reducing them to their root word will help in reducing the total number of unique words in our data without losing a significant amount of information. We will use PorterStemmer() function from nltk for stemming the words. Now we can normalize the tokenized tweets.

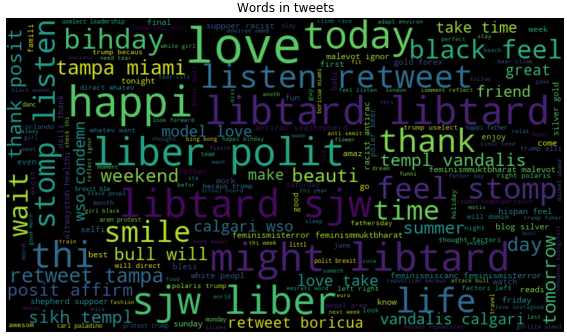


**iii) Data Visualization**

After data cleaning now we can explore the cleaned data. We will explore and visualize the following things:

1. All words used in the tweets:

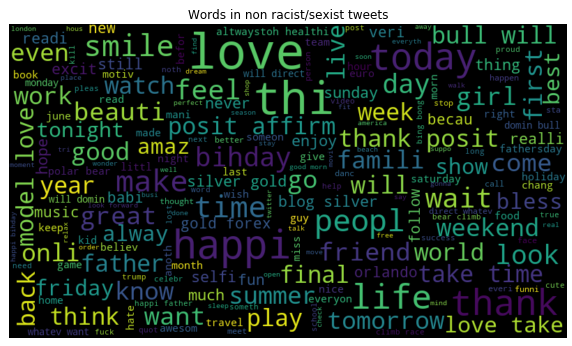
We will see the all words by plotting wordcloud. A wordcloud is a visualization in which the most frequent words appear in large size and the less frequent words appear in smaller sizes. The wordcloud for all the words we get,



From the wordcloud, we see that love, thank, listen etc. occurred more frequently than summer, weekend, model etc.

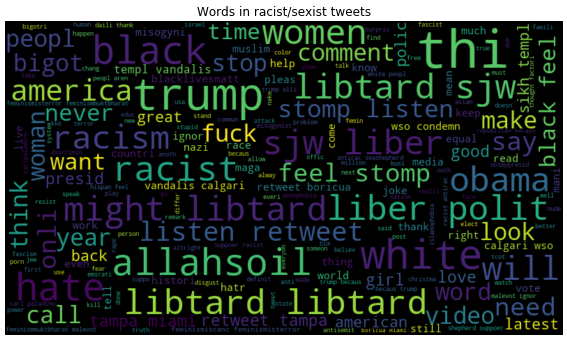
1. Words in non-racist/sexist tweets

Now we will see the words associated with non-racist/sexist tweets or normal tweets.



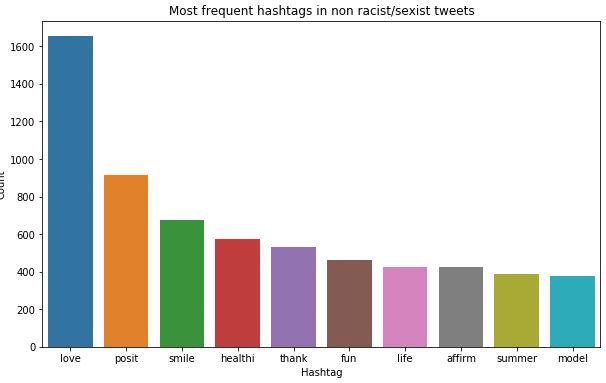
Words in racist/sexist tweets

We will check the words associated with racist/sexist tweets.



1. Hashtags in Non racist/sexist tweets:

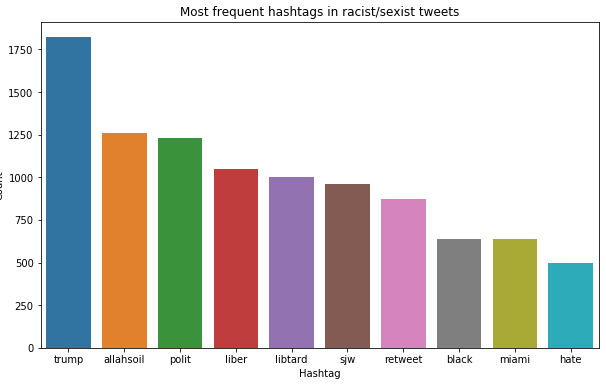
Hashtags in twitter represent the ongoing trends on twitter at any particular point in time. So, we will check whether these hashtags add any value to our sentiment analysis task. First we will check the non-racist/sexist or normal tweet hashtags.



We can see the most frequent ten hashtags in normal tweets. These hashtags don’t have any negative terms.

1. Hashtags in Racist/sexist tweets:

Now let’s check the hashtags associated with negative terms.



From the bar plot , we can see that most of the hashtags have negative terms and a few have neutral terms. We will keep these hashtags as they contain valuable information.

**Implementation**

To analyze a preprocessed data, it needs to be converted into features. Depending upon the usage, text features can be constructed using assorted techniques – TfidfVectorizer and CountVectorizer.

1. TfidfVectorizer: The [TfidfVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html) (Term frequency-inverse document frequency) will tokenize documents, learn the vocabulary and inverse document frequency weightings (downscales words that appear a lot across documents) and allow users to encode new documents.
2. CountVectorizer: The [CountVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.

Algorithms we will use are:

1. Logistic Regression
2. K Nearest Neighbors (KNN)
3. Decision Tree
4. Random Forest
5. Support Vector Machine (SVM)
6. 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. Logistic regression can be considered as a special case of linear regression when the outcome variable is categorical, where we are using log of odds as the dependent variable.
7. K Nearest Neighbors (KNN) is non parametric method for solving both regression and classification type of problem where k is the closest observed observation or neighbors. It’s a lazy learning algorithm where all computation is differed. Inputs in this algorithm can be both categorical and numerical but the outputs are categorical values.
8. Decision Trees are a type of Supervised Machine Learning where the data is continuously split according to a certain parameter. It can be used to solve both regression and classification problems. Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.
9. Random Forest is a versatile machine learning algorithm capable of performing both regression and classification tasks. It is a kind of ensemble learning method, where a few weak models combine to form a powerful model. In Random Forest, we grow multiple trees as opposed to a decision single tree. To classify a new object based on attributes, each tree gives a classification and we say the tree “votes” for that class. The forest chooses the classification having the most votes.
10. Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for solving both classification and regression type problems. 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 the 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 differentiate the two classes.

First we will apply CountVectorizer method for all the algorithms. We will check the F1 score, Recall and Precision of all the algorithms.

1. For Logistic Regression F1 score, Recall and Precision are 0.93, 0.93 and 0.93.
2. For KNN F1 score, Recall and Precision are 0.92, 0.92 and 0.92.
3. For Decision Tree F1 score, Recall and Precision are 0.93, 0.93 and 0.93.
4. For Random Forest F1 score, Recall and Precision are 0.94, 0.94 and 0.94.
5. For SVM F1 score, Recall and Precision are 0.37, 0.52 and 0.75.

Now we will apply TfidfVectorizer for all the algorithms and check the F1 score, Recall and Precision of all the algorithms.

1. For Logistic Regression F1 score, Recall and Precision are 0.93, 0.93 and 0.93.
2. For KNN F1 score, Recall and Precision are 0.92, 0.92 and 0.92.
3. For Decision Tree F1 score, Recall and Precision are 0.93, 0.93 and 0.93.
4. For Random Forest F1 score, Recall and Precision are 0.93, 0.93 and 0.93.
5. For SVM F1 score, Recall and Precision are 0.37, 0.52 and 0.75.

We get better performance for Random Forest by using [CountVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) technique. So, we will use Random Forest with [CountVectorizer](http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) for this project.

For the model we get,

True Positive Rate (TPR) : 99.79

True Negative Rate (TNR) : 97.84

False Negative Rate (FNR) : 0.21

False Positive Rate (FPR) : 2.16

After completion of the model we will use HTML, CSS and Flask for building a web application.



**Experimental Results**

We will use some comments and observe the outputs for them.

|  |  |  |
| --- | --- | --- |
| **Comment** | **Prediction** | **Expectation** |
| I like to watch movies | Positive | Positive |
| The Negroes are not so smart | Negative | Negative |
| Thank you guys | Positive | Positive |
| Gender discrimination should not be eradicated | Negative | Negative |
| She is so ugly | Negative | Negative |

For a positive comment

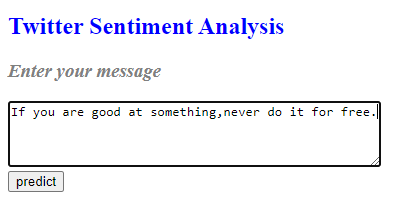


Figure: Positive comment input



Figure: Positive comment prediction

For a negative comment

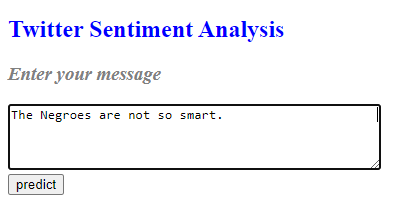


Figure: Negative comment input

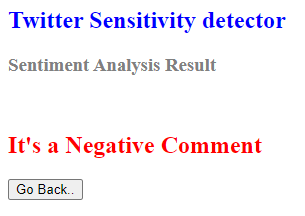
****

Figure: Negative comment prediction

From the test results, we can see that our model is predicting positive comment as positive and negative comment or message as negative. So the model is working as we expected.

**Conclusion and Future Scope**

Our Twitter Sentiment Analysis model is detecting racist/sexist tweets as we expected. So we can say our model is working fine. But still we can try other things too.

1. We can try model ensembling.
2. Parts-of-Speech tagging can be used to create new features.
3. We can use lemmatization as it might help in getting rid of unnecessary words.
4. Also a better dataset with proper labelling will give more accurate result.

We can use this model in various websites for detecting negative comments or messages or social media monitoring. Also it can be used for brand monitoring, customer service and market research which are quite similar to our model.

**References**

1. <https://www.kaggle.com/arkhoshghalb/twitter-sentiment-analysis-hatred-speech>
2. <https://pythonprogramming.net/stemming-nltk-tutorial/>
3. <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html>
4. <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>