Analysing human sentiment using machine learning-based techniques: A case study on COVID-19 pandemic

A Project Report Submitted in Partial Fulfilment of the Requirements for the Award of the Degree of

Bachelor of Technology
in
Computer Science & Engineering

By

Purnendu Das
Roll no- 11100116031

Under the Supervision of
Dr. Madhulina Sarkar
Assistant professor,
Head of the Computer Science and Engineering Department

Government College of Engineering and Textile Technology,
Berhampore, Murshidabad, India.
July 2020

Berhampore – 742101
Murshidabad , West Bengal

Phone No. – 250142
Fax No. – 250142

GOVT. COLLEGE OF ENGG. TEXTILE TECHNOLOGY, BERHAMPORE

(Govt. of West Bengal)

Certificate of Approval

This is to certify that the project report on “Analysing human sentiment using machine learning-based techniques: A case study on COVID-19 pandemic" is a record of bonafide work, carried out by Mr. PURNENDU DAS under my guidance and supervision.

In my opinion, the report in its present form is in conformity as specified by Government College of Engineering & Textile Technology, Berhampore, W.B and as per regulations of the Maulana Abul Kalam Azad University of Technology. To the best of my knowledge the results presented here are original in nature and worthy of incorporation in project report for the Bachelor of Technology in Computer Science and Engineering.

____
Signature of the Project Supervisor (Head of the CSE Dept.)

ACKNOWLEDGEMENT

With great pleasure I would like to express my profound gratitude and indebtedness to **Dr. Madhulina Sarkar**, Assistant Professor and Head of the Computer Science & Engineering Department, Govt. College of Engineering & Textile Technology, Berhampore, West Bengal, for her continuous guidance, valuable advice and constant encouragement throughout the project work. Her valuable and constructive suggestions at many difficult situations are immensely acknowledged. I am in short of words to express his contribution to this thesis through criticism, suggestions and discussions.

PURNENDU DAS,
11100116031

Abstract

Sentiment analysis is a very interesting filed for research-based project work. It has significant importance in e-commerce and many other public fields. There are many resources available online to collect public data. Sentiment analysis from tweets has attracted more attention nowadays due to the COVID-19 pandemic situation. Twitter is a part of our regular life. Twitter is used as a micro-blogging site where people can post and interact with each other, it is the best platform to collect the audience's emotions. Sentiment Classification seeks to identify a piece of text according to its author's general feeling toward their subject, be it positive or negative. Traditional machine learning techniques have been applied to this problem with reasonable success. In this project work, we present data pre-processing with help of predefined lexicons and labelled with sentiment polarity and build a supervised machine learning model from word count vector with its polarity. We have used several supervised techniques like the Naive Bayes technique, Random forest technique, Decision Tree technique and Logistic regression technique. We have analysed the outcome result of every method from two different datasets. Dealing with tweets many circumstances were present like Emojis, Emoticons, Digits, Website links, many special characters in a tweet text. It is very difficult to build a high accuracy machine learning model with those things, in this project we have removed the emojis, emoticons, website links, digits, and special characters from the dataset. We got the highest accuracy on the sentiment balanced dataset with the Logistic regression approach.

<u>CONTENTS</u>

1. Introduction

2. Objective

3. Literature Review

4. Methodology

5. Implementation Details and Results

6. Conclusion and future work

7. References

1.Introduction

Sentiment analysis or opinion mining is one of the major tasks of NLP (Natural Language Processing). Sentiment analysis deals with identifying and classifying opinions or sentiments expressed in the source text. Social media is generating a vast amount of sentiment rich data in the form of tweets, status updates, blog posts, etc. Sentiment analysis of this user generated data is very useful in knowing the opinion of the crowd. Human opinions have very important roles in everywhere. In ecommerce, shopper must want to know the buyer's opinion and the sentiment. And also, many other aspects.

Twitter sentiment analysis is difficult compared to general sentiment analysis due to the presence of slang words and misspellings. The maximum limit of characters that are allowed in Twitter is 140. And also, it has many website links, emojis, emoticons, special characters. Twitter API also have limitations. Knowledgebase approach and Machine learning approach are the two strategies used for analysing sentiments from the text. In this project, we try to analyse twitter posts using the lexicons sentiment polarity & Machine Learning approaches. By doing sentiment analysis in a specific domain, it is possible to identify the effect of domain information in sentiment classification. In this project our domain is COVID-19 pandemic situation and targeted tweets related to **#lockdown #india.** We present a new feature lexicons sentiment polarity and word count vector for classifying the tweets as positive or negative depends on the polarity.

We examine the effectiveness of applying machine learning techniques to the sentiment classification problem. Our analysis helps concerned organizations to find opinions of people about current pandemic situation from their tweets, if it is positive or negative. The challenging aspect in sentiment analysis is an opinion word which is considered as a positive in one situation may be considered as negative in another situation. The traditional text processing with porter stemming technique may change the meaning of the original sentence. Whereas we can use word lemmatization for better text processing. There are different algorithms that can be used in the stemming process, but the most common in English is Porter stemmer. The rules contained in this algorithm are divided in five different phases numbered from 1 to 5. The purpose of these rules is to reduce the words to the root. Lemmatization is the key to this methodology is linguistics. To extract the proper lemma, it is necessary to look at the morphological analysis of each word. This requires having dictionaries for every language to provide that kind of analysis.

In this project report we have demonstrate how the dataset is pre-processed and lemmatizing. How to determine the sentence polarity with help of lexicons and update the dataset with the sentence polarity. How much a machine learning model's accuracy depends on the sentiment polarity of the maximum number of the data. Improvement of different machine learning models. And select the best model among them. Finally result analysis and practical implementation.

2. Objective

This project is implemented of sentiment classification of a tweet and build a best model that can helps to predict the sentiment with high accuracy. In this project our aims are:

- Build supervised unbiased model.
- Choose the best model.
- Increase the model's precision value.
- Increase the model's accuracy.
- Make a record of the model with real example text.

3. Literature Review

Twitter sentiment classification, which identifies the sentiment polarity of short, informal tweets, has attracted increasing research interest (Jiang et al., 2011; Hu et al., 2013) in recent years. Generally, the methods employed in Twitter sentiment classification follow traditional sentiment classification approaches.

The lexicon-based approaches (Turney, 2002; Ding et al., 2008; Taboada et al., 2011; Thelwall et al., 2012) mostly use a dictionary of sentiment words with their associated sentiment polarity, and incorporate negation and intensification to compute the sentiment polarity for each sentence (or document).

The learning-based methods for Twitter sentiment classification follow Pang et al. (2002)'s work, which treat sentiment classification of texts as a special case of text categorization issue.

Many studies on Twitter sentiment classification (Pak and Paroubek, 2010; Davidov et al., 2010; Barbosa and Feng, 2010; Kouloumpis et al., 2011; Zhao et al., 2012) leverage massive noisy-labelled tweets selected by positive and negative emoticons as training set and build sentiment classifiers directly, which is called distant supervision (Go et al., 2009). Instead of directly using the distant supervised data as training set, Liu et al. (2012) adopt the tweets with emoticons to smooth the language model and Hu et al. (2013) incorporate the emotional signals into an unsupervised learning framework for Twitter sentiment classification. Many existing learning-based methods on Twitter sentiment classification focus on feature engineering. The reason is that the performance of sentiment classifier being heavily dependent on the choice of feature representation of tweets. The most representative system is introduced by Mohammad et al. (2013), which is the state-of-the art system (the top-performed system in SemEval 2013 Twitter Sentiment Classification Track) by implementing a number of hand-crafted features. Unlike the previous studies, we focus on learning discriminative features automatically from massive distant-supervised tweets.

4. Methodology

In this project work we have made 5 different phases. Which are

1. Data collecting through twitter API.
2. Data cleaning.
3. Data pre-processing with help of POS tagging
4. Determine the sentiment polarity via Lexicons and make labelling.
5. Build supervised model

Tools are used:

1. Python libraries:
   I. NLTK [NLP tools]
   II. TextBlob [Text processing tools]
   III. SKlearn [Machine Learning tools]
   IV. Pandas [Dataset handling tool]
   V. Matplotlib [Data visualization tools]
   VI. Tweepy [Twitter API tools]
   VII. Re [Regular expression tools]
   VIII. Numpy [array & math tools]
2. Twitter API

4.1. Data collect

We have created a developer account on twitter. Created a new App on twitter, create API keys and other token keys, etc.

On python with help of the tweepy library, authenticate with the twitter API. Make a search query with searching term “lockdown” and “india”. Also applied the retweet filter with extended tweet mode. Max tweet 30,000. Per iterartion. We have collected 1.2L tweets.

After getting the all tweets, created a pandas dataframe and store the dataframe into dataset.csv file.

4.2. Data cleaning

Created list of happy emoticons and sad emoticons, regular expression pattern of special characters, regular expression pattern of name tagging, regular expression pattern of emojis, regular expression pattern of Unicode characters and digits, regular expression pattern of website URL.

Now we have passed every tweet in that process where every tweet cleaned the emoji, emoticon, special character, digit, tagging, URL from the text.

Next process is tokenizing the sentence. And removed the non-English words. Making filtered sentence of pure English words only from the original tweet text.

4.3. Data pre-processing with help of POS tagging

After getting the filtered sentences now process to making the label with sentiment polarity. Instead of traditional porter stemming we have used lemmatizing.

4.3.1. Porter's Stemmer algorithm

It is one of the most popular stemming methods proposed in 1980. It is based on the idea that the suffixes in the English language are made up of a combination of smaller and simpler suffixes.

Example: EED -> EE means "if the word has at least one vowel and consonant plus EED ending, change the ending to EE" as 'agreed' becomes 'agree'.

CONNECT
CONNECTIONS------> CONNECT
CONNECTED------> CONNECT
CONNECTING------> CONNECT
CONNECTION------> CONNECT

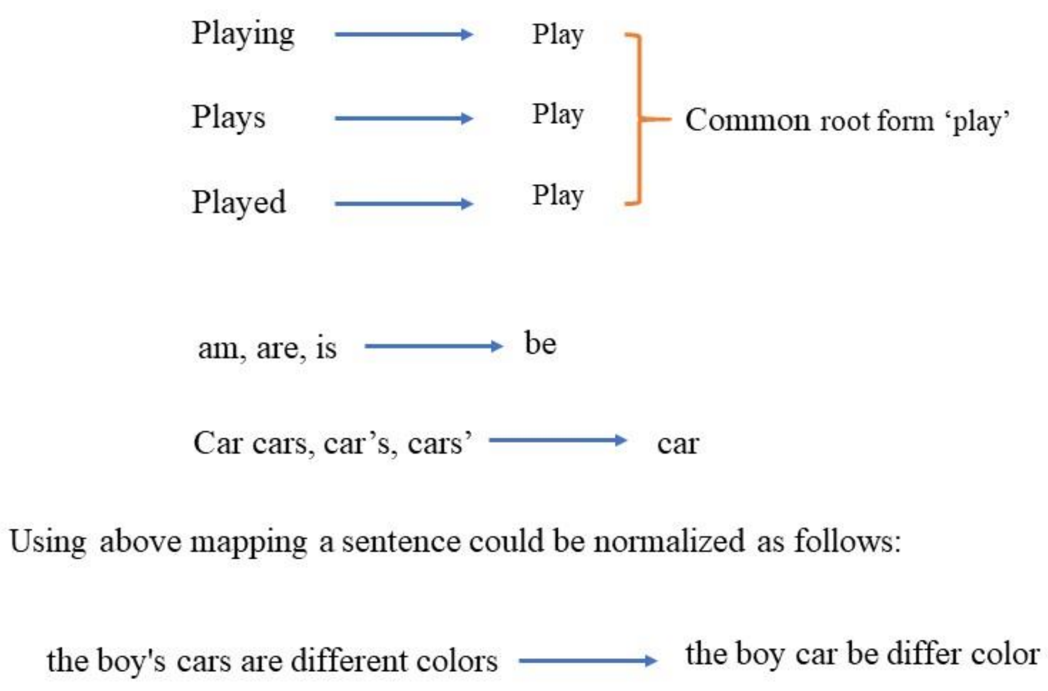
Advantage: It produces the best output as compared to other stemmers and it has less error rate.

Limitation: Morphological variants produced are not always real words.

4.3.2. Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words. So, it links words with similar meaning to one word.

One major difference with stemming is that lemmatize takes a part of speech parameter, "pos" If not supplied, the default is "noun."

Examples of lemmatization:

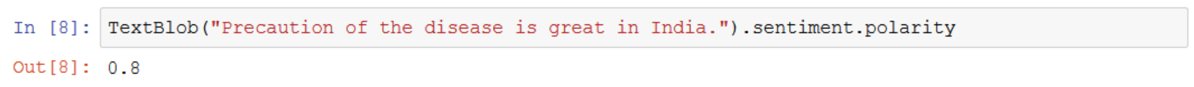

Playing
Play
Plays
Play
Played
Play
Common root form 'play'
am, are, is
be
Car cars, car's, cars'
car
Using above mapping a sentence could be normalized as follows:
the boy's cars are different colors
the boy car be differ color

4.4. Lemmatizing and find polarity of a sentence

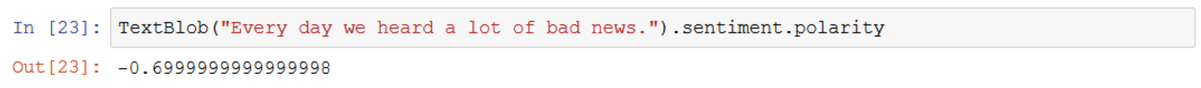
Create a function and find out each word's POS [past of speech] tag with help of nltk.pos_tag() function.
According the POS tag get every word's lemmatized word by NLTK library.
Replace the sentence with lemmatized sentence.

We have used TextBlob lexicons analyse method for determine the sentiment polarity.
Store the polarity in the data set.

TextBlob sentiment polarity Examples:



```
In [8]: TextBlob("Precaution of the disease is great in India.").sentiment.polarity
Out[8]: 0.8
```



```
In [23]: TextBlob("Every day we heard a lot of bad news.").sentiment.polarity
Out[23]: -0.6999999999999998
```

We have collected the polarity and labelled the dataset with 0 and 1.

0 = Negative [polarity less than 0]

1 = Positive Sentiment [polarity greater than equal to 0]

***in TextBlob polarity 0 means neutral sentiment, but we are building a binary classification model so we have decided to consider the neutral polarity with positive sentiment.*

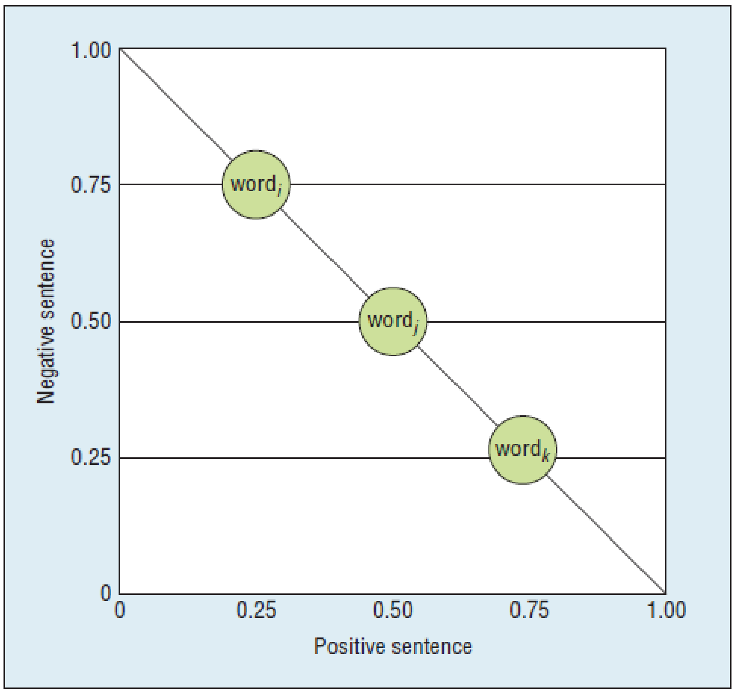


Figure: Relationships between words and sentiment of that sentences, assuming there are two types of sentiment sentences: positive and negative. Words i, j, and k are classified as objective according to TextBlob. Here, word k has a stronger positive sentiment tendency than words i and j, and word i has a stronger negative sentiment tendency than words j and k.

4.5.Build Supervised Machine Learning Model

We have 95,322 tweets. Filter the tweets with at least of 3 words present on the tweet. Number of tweets has decreased and we got 90,793 tweets. Checked the dataset's sentiments total counts. And plot a bar chart from the data.

```
In [11]: tweets.sentiment.value_counts()

Out[11]: 1    61255
         0    29538
         Name: sentiment, dtype: int64
```

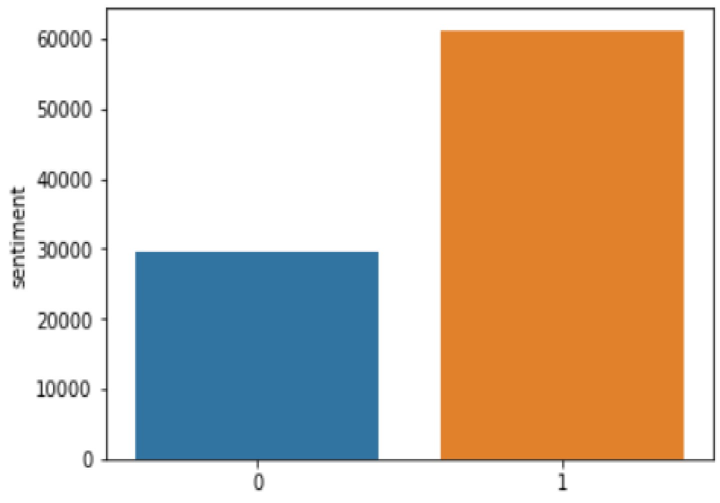
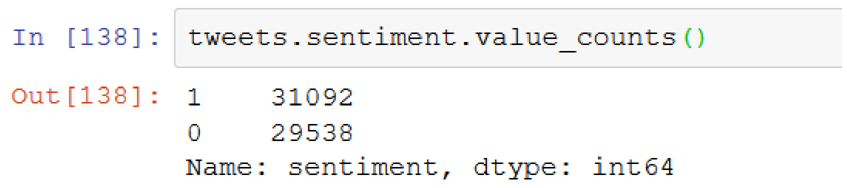


Figure: Bar chart of value counts of the sentiments
0 = Negative Tweet, 1 = Positive Tweet

It is a positive biased dataset. We can't make a good machine learning model from that dataset. So, we have reduced many positive sentiment tweets of polarity 0 to 0.07. Our motive to create an unbiased dataset where positive and negative sentiment will be balanced. 30,163 tweets are removed from the dataset to balance the dataset.



```
In [138]: tweets.sentiment.value_counts()

Out[138]: 1    31092
          0    29538
          Name: sentiment, dtype: int64
```

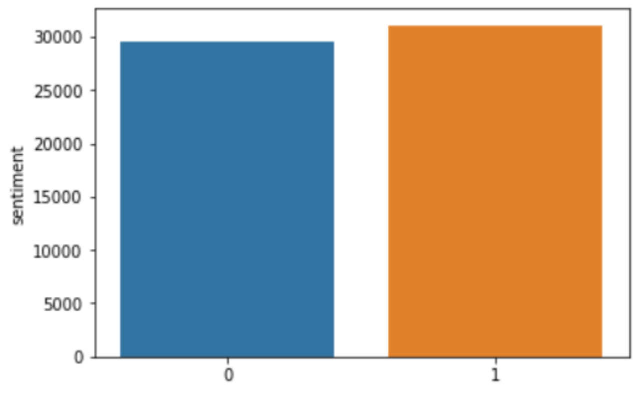


Figure: Bar chart of value counts of the sentiments
0 = Negative Tweet, 1 = Positive Tweet

Now we got almost balanced dataset. Which is unbiased into a particular sentiment. Collect all unique words of the whole data sets which has at least used frequency of 3. We get 8,687 words.

Create a word count vector format of the whole dataset. Columns are the unique words and rows are the word count number in that tweet.

Make the word count vector as X. And sentiment labelled column as y.

Shape of the X is (60630, 8687) and shape of the y is (60630, 1).

Now time for splitting the dataset into training and testing datasets. We have splitted the whole datasets into 75% and 25% for training and testing.

We have built 4 different supervised models.

4.5.1. Random Forest Model:

Working of Random Forest Algorithm

We can understand the working of Random Forest algorithm with the help of following steps –

- First, start with the selection of random samples from a given dataset.
- Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.
- In this step, voting will be performed for every predicted result.
- At last, select the most voted prediction result as the final prediction result.

  The following diagram will illustrate its working –

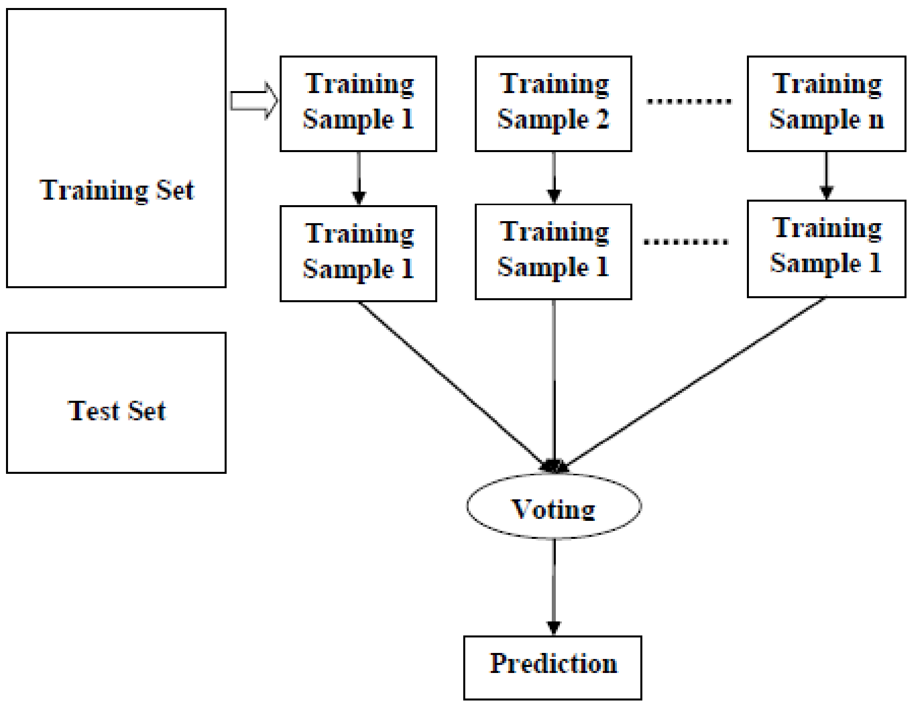


Fig. *Random forest algorithm working diagram*

4.5.2. Decision tree Model

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the CART algorithm**,** which stands for Classification and Regression Tree algorithm.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Below diagram explains the general structure of a decision tree:

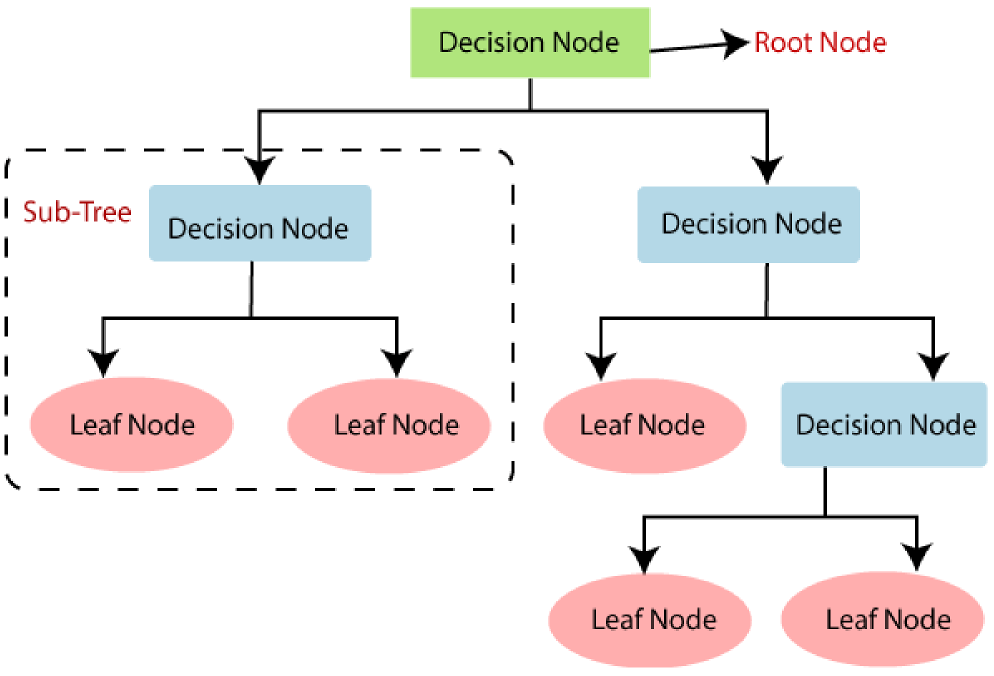


Fig. Decision tree algorithm diagram

4.5.3. Logistic regression model

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability. We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the '**Sigmoid function**' or also known as the 'logistic function' instead of a linear function. The hypothesis of logistic regression tends it to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

$$0 \leq h_\theta(x) \leq 1$$

Logistic regression hypothesis expectation

In order to map predicted values to probabilities, we use the Sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

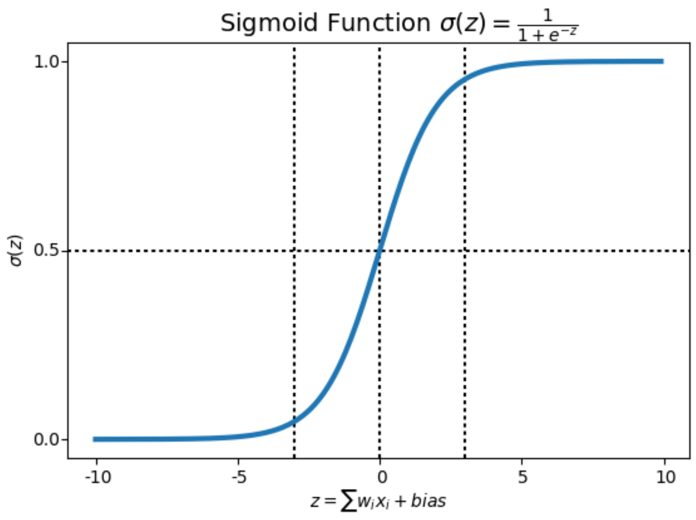


Fig. Sigmoid Function Graph

Formula of a sigmoid function: $f(x) = \dfrac{1}{1+\mathrm{e}^{-(x)}}$

Hypothesis Representation
When using linear regression, we used a formula of the hypothesis i.e.
$h\Theta(x) = \beta_0 + \beta_1 X$
For logistic regression we are going to modify it a little bit i.e.
$\sigma(Z) = \sigma(\beta_0 + \beta_1 X)$

We have expected that our hypothesis will give values between 0 and 1.
$Z = \beta_0 + \beta_1 X$
$h\Theta(x) = \text{sigmoid}(Z)$
i.e. hΘ(x) = 1/(1 + e^-(β₀ + β₁X)

$$h\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Fig. The Hypothesis of logistic regression

Cost Function

We learnt about the cost function $J(\theta)$ in the Linear regression, the cost function represents optimization objective i.e. we create a cost function and minimize it so that we can develop an accurate model with minimum error.

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2.$$

Fig. The Cost function of Linear regression

If we try to use the cost function of the linear regression in ‘Logistic Regression’ then it would be of no use as it would end up being a non-convex function with many local minimums, in which it would be very difficult to minimize the cost value and find the global minimum.

For logistic regression, the Cost function is defined as:
$-\log(h\theta(x))$ if y = 1
$-\log(1-h\theta(x))$ if y = 0

$$Cost(h_\theta(x), y) = \begin{cases} -log(h_\theta(x)) & \text{if } y = 1 \\ -log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Fig. Cost function of Logistic Regression

The above two functions can be compressed into a single function i.e.

$$J(\theta) = -\frac{1}{m}\sum\left[y^{(i)}\log(h\theta(x(i))) + \left(1 - y^{(i)}\right)\log(1 - h\theta(x(i)))\right]$$

Above functions compressed into one cost function

4.5.4. Naïve Bayes (Multinomial NB)

Naive Bayes are a group of supervised machine learning classification algorithms based on the **Bayes theorem**. It is a simple classification technique, but has high functionality. They find use when the dimensionality of the inputs is high. Complex classification problems can also be implemented by using Naive Bayes Classifier.

The Formula For Bayes' Theorem Is

$$P(A|B) = \frac{P(A \bigcap B)}{P(B)} = \frac{P(A) \cdot P(B|A)}{P(B)}$$

where:

$P(A) =$ The probability of A occurring

$P(B) =$ The probability of B occurring

$P(A|B) =$ The probability of A given B

$P(B|A) =$ The probability of B given A

$P\left(A \bigcap B\right)) =$ The probability of both A and B occurring

Combining probability distribution of P with fraction of documents belonging to each class.

For class **j**, word **i** at a word frequency of **f**:

$$Pr(j) \propto \pi_j \prod_{i=1}^{|V|} Pr(i|j)^{f_i}$$

In order to avoid underflow, we will use the sum of logs:

$$Pr(j) \propto \log(\pi_j \prod_{i=1}^{|V|} Pr(i|j)^{f_i})$$

$$Pr(j) = \log \pi_j + \sum_{i=1}^{|V|} f_i \log(Pr(i|j))$$

One issue is that, if a word appears again, the probability of it appearing again goes up. In order to smooth this, we take the log of the frequency:

$$Pr(j) = \log \pi_j + \sum_{i=1}^{|V|} log(1 + f_i) \log(Pr(i|j))$$

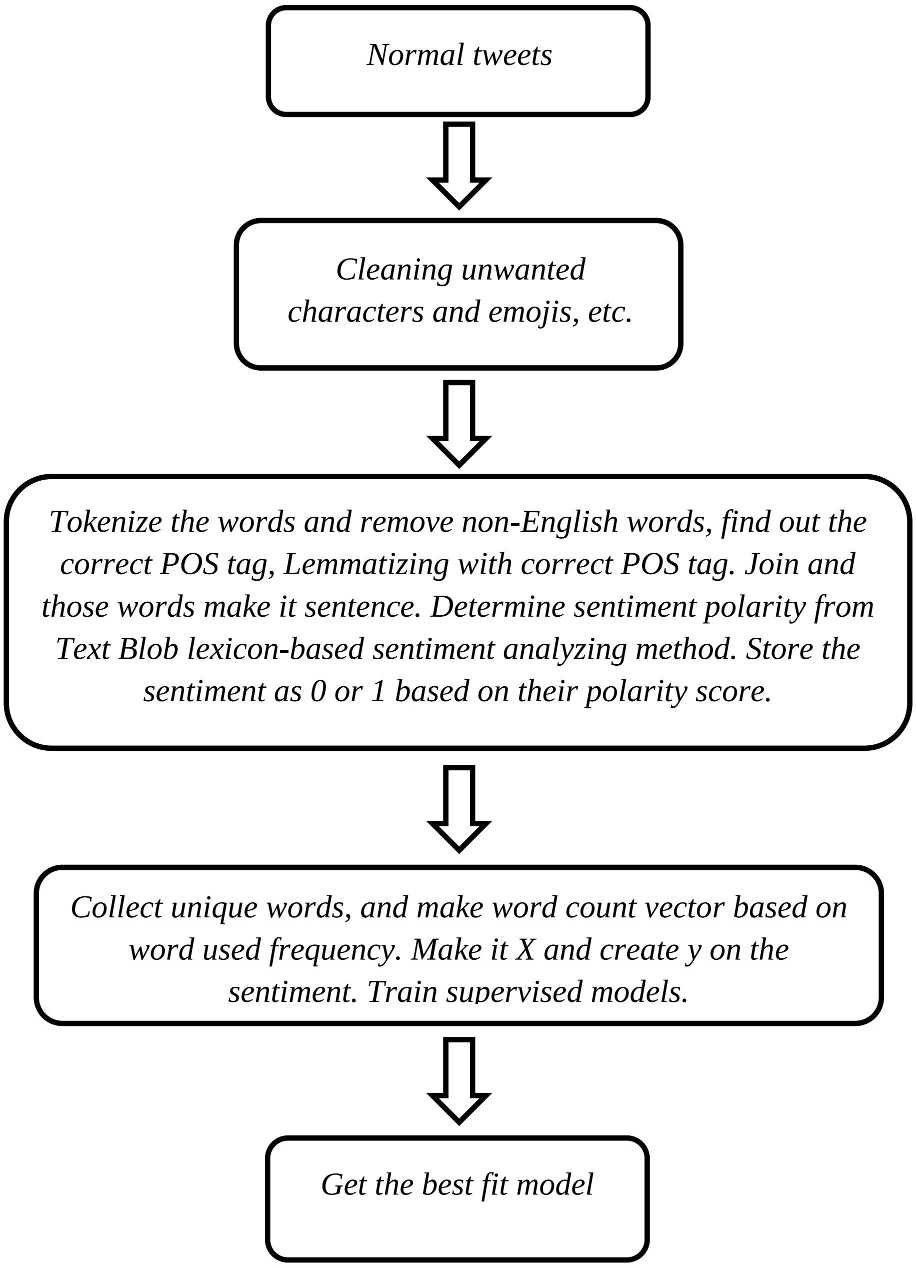
Also, in order to take stop words into account, we will add an Inverse Document Frequency (IDF)weight on each word:

$$t_i = \log\left(\frac{\sum_{n=1}^{N} doc_n}{doc_i}\right)$$

Optimal Multinomial Naïve Bayes model is:

$$Pr(j) = \log \pi_j + \sum_{i=1}^{|V|} log(1 + f_i) \log(Pr(i|j))$$

4.6. Flow Chart of our methodology

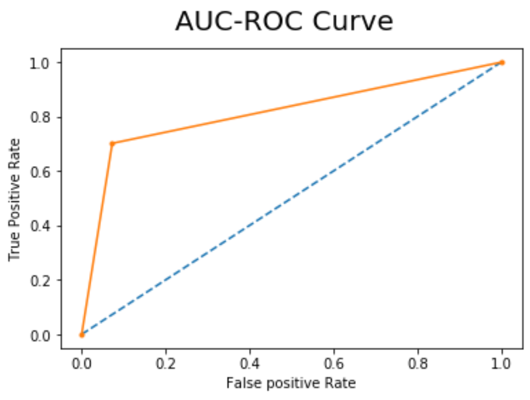

Normal tweets
Cleaning unwanted characters and emojis, etc.
Tokenize the words and remove non-English words, find out the correct POS tag, Lemmatizing with correct POS tag. Join and those words make it sentence. Determine sentiment polarity from Text Blob lexicon-based sentiment analyzing method. Store the sentiment as 0 or 1 based on their polarity score.
Collect unique words, and make word count vector based on word used frequency. Make it X and create y on the sentiment. Train supervised models.
Get the best fit model

5. Implementation Details and Results

5.1. Trained the dataset among 5 different algorithms and checked the results as below:

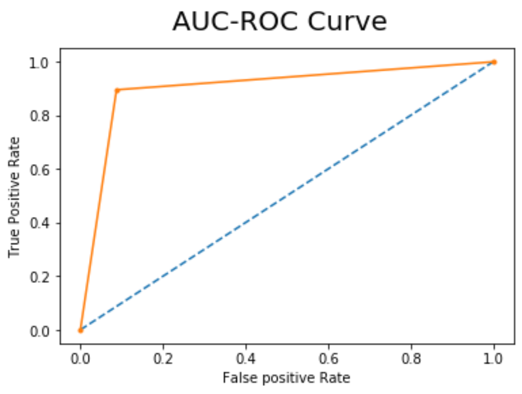
5.1.1. Random Forest algorithm model

Accuracy: 0.8357126140884624
Precision: 0.8681397006414825
Recall: 0.7020172910662824
Log loss: 5.6743199051607505
F1 Score: 0.7762906309751434
AUC: 0.8145626250445674



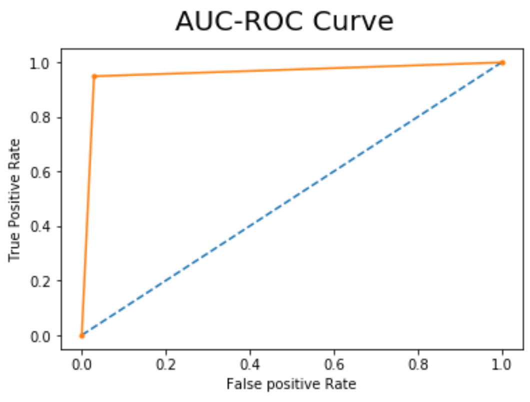
5.1.2. Decision Tree algorithm model

Accuracy: 0.9056868710507839
Precision: 0.875
Recall: 0.8956772334293948
Log loss: 3.2575016142706557
F1 Score: 0.8852178866419823
AUC: 0.904103392128409



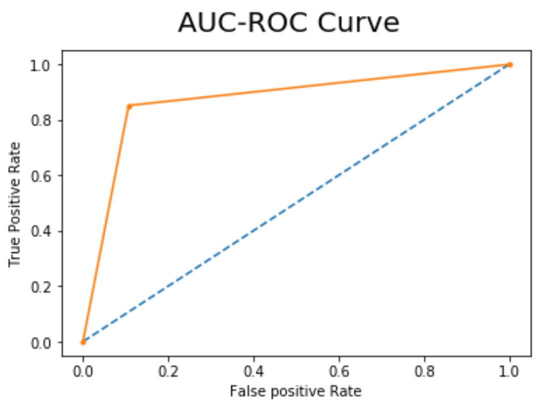
5.1.3. Logistic Regression model

Accuracy: 0.9620283018867924
Precision: 0.9570049722140976
Recall: 0.9492312155497534
Log loss: 1.3115098514131467
F1 Score: 0.9531022429362075
AUC: 0.9600119916413579



5.1.4. Naïve Bayes model (Multinomial)

Accuracy: 0.8761993915282004
Precision: 0.8445714285714285
Recall: 0.8518731988472622
Log loss: 4.275972432344669
F1 Score: 0.8482065997130559
AUC: 0.87235109902962



5.2. Real examples:

```
pred_text("i am mentally disturbed during the quarantine")
```

```
Logistic Regression Prediction : NEGATIVE
```

```
pred_text("India is fighting the pandemic situation greatly.")
```

```
Logistic Regression Prediction : POSITIVE
```

```
pred_text("in this lockdown period i have learned so many new things")
```

```
Logistic Regression Prediction : POSITIVE
```

```
pred_text("in this quarantine period, our environment is curing.")
```

```
Logistic Regression Prediction : POSITIVE
```

6. Conclusion and future work

6.1. We have experiment with different models and we got the best results on the **Logistic regression model**. With the accuracy of 96% which is best accuracy among the other 3 models. And we have also gotten the minimum log loss of 1.31 with a good precision value (0.95). Our project objectives have successfully fulfilled.

6.2. We have planned to go through with different methods with the dataset. In the Deep Learning methods likes BERT, LSTM will be tested with the dataset for more accuracy.

7. References

7.1. Datacamp : https://www.datacamp.com/community/tutorials/stemming-lemmatization-python

7.2. Sentiment analysis on Wikipedia : https://en.wikipedia.org/wiki/Sentiment_analysis

7.3. Alexander Pak, Patrick Paroubek , "Twitter as a Corpus for Sentiment Analysis and Opinion Mining" , Universit´e de Paris-Sud, Laboratoire LIMSICNRS, Bˆatiment 508, F-91405 Orsay Cedex, France alexpak@limsi.fr,pap@limsi.fr

7.4. Application of sentimental analysis in adaptive user interfacesBadruddin Kamal,Abu M. Hammad Ali, Dr. Mumit Khan,BRAC University,Dhaka, Bangladesh, Fall"2011

7.5. Sentiment Classification using Machine Learning Techniques Suchita V Wawre1, Sachin N Deshmukh2

7.6. Knowledge-Based Approaches to Concept-LevelSentimentAnalysisUsing Objective Words in SentiWordNet to Improve Word-of- Mouth Sentiment Classification Chihli Hung and Hao-Kai Lin, Chung Yuan Christian University

7.7. Using Emoticons to reduce Dependency in Machine Learning Techniques for Sentiment Classification Jonathon Read Department of Informatics University of Sussex United Kingdom j.l.read@sussex.ac.uk

7.8. Learning Sentiment-SpecificWord Embedding for Twitter Sentiment Classification Duyu Tang†, Furu Wei‡ , Nan Yang\, Ming Zhou‡, Ting Liu†, Bing Qin††Research Center for Social Computing and Information Retrieval Harbin Institute of Technology, China ‡Microsoft Research, Beijing, China University of Science and Technology of China, Hefei, China

7.9. Sentiment Analysis-An Objective View Soumi Sarkar Taniya Seal B. Tech Student B. Tech Student Department of Computer Science and Engineering Department of Computer Science and Engineering University of Calcutta University of Calcutta Prof. Samir K. Bandyopadhyay Professor Department of Computer Science & Engineering University of Calcutta

7.10. Twitter Sentiment Classification using Distant Supervision by Alec Go, Richa Bhayani, Lei Huang, Stanford University, Stanford, CA 94305

7.11. *Sentiment Lexicon Interpolation and Polarity Estimation of Objective and Out-Of-VocabularyWords to Improve Sentiment Classification on Microblogging Yongyos Kaewpitakkun, Kiyoaki Shirai, Masnizah Mohd Japan Advanced Institute of Science and Technology 1-1, Asahidai, Nomi City, Ishikawa, Japan 923-1292*

7.12. *Barbosa and J. Feng. 2010. Robust sentiment detection on twitter from biased and noisy data. In Proceedings of Coling.*