Implementing Sentiment Analysis on Real-Time Twitter Data

Hisham Ahmed Sheikh, Dr. Jitendra Jaiswal

M. Tech Student, Assistant Professor,
Department of Computer Science and Engineering (Specialization),
School of Engineering and Technology, Jain University, Bengaluru-560027, Karnataka, India.

Abstract : This study has been undertaken to investigate the application of Sentiment Analysis on real time Twitter data collected through Twitter API. Sentiment analysis is a progressive field of natural language processing. It is a way to detect the attitude, state of mind, or emotions of the person towards a product, service, movie, etc. by analyzing the opinions and reviews shared on social media, blogs and so on. Various social media platforms such as Facebook, Twitter and so on allow people to share their views with other people. Twitter has become the most popular social media platform that allows users to share information by way of the short messages called tweets on a real-time basis. These tweets consist of user opinions towards a particular topic, trend, or issue. These Tweets are first extracted in real time through application programming interfaces (APIs) using hashtags and keywords, about political figures such as Donald Trump, Narendra Modi, etc. They are then categorized based on their subjectivity and polarity. Machine learning algorithms such as Naïve Bayes and Random Forest Classifier are then used to accurately predict these Tweets as negative or non-negative (positive or neutral). This will give us an understanding on general public reception towards these figures. Our experimental results show that it is possible to process data in real-time, and obtain information continuously, albeit with varying classification accuracies due to real- time data collection.

***Index Terms* - *Sentiment Analysis, Real-Time, Twitter, Random Forest, Naïve Bayes*.**

I. INTRODUCTION

Sentiment analysis alludes to the utilization of natural language processing (NLP), text analysis, etymology, and insights to reliably set up extricate measure and study emotive states as well as natural or individual information or opinion. It is broadly applied to the voice of the customer or client materials like audits, reviews, reactions on the web and web-based life and social insurance materials for applications like customer administration to identify sentiments, feelings, attitude, and state of mind of people towards a product or service. These sentiments are then classified as positive or negative based on the data collected. With the expansion of the internet, a huge amount of organized and unorganized data is produced through various social media platform that allows people to share their opinion, comment, feedback, review, thoughts, and experiences with the world. Among all the social media platforms such as Twitter, Facebook, YouTube, and so on, Twitter becomes most popular social media platform these days to publically share thoughts, feelings, views, and opinions with the world. It provides the facility that allows users to share information by way of the short messages called tweets on a real-time basis. Sentiment analysis helps us in automatically transforming an unorganized large amount of data into an organized form in few minutes. Researchers, companies, governments implement it for further advertising or marketing of their products and also improve their standing in society.

In this paper, a huge amount of tweets collected in real-time are analysed for their subjectivity and polarity and then subsequently classified. In our proposed work, we employ sentiment analysis using a set of packages supported by Python language extract tweets using Tweepy library and carry out the sentiment analysis for tweets on political figures such as US President Donald Trump, and Indian Prime Minister Narendra Modi. We then analyze the sentiments of the public towards that public figure. The system will perform analysis in six phases which are data collection, data pre-processing, feature extraction, model preparation, training and testing data, results presentation. We identify the subjectivity and polarity of the tweets and compute both the subjectivity and polarity scores.

Twitter [1] is used as a data source to collect real-time data in our system on any topic through twitter application programming interfaces (APIs) [2] using hashtags or search keywords. The extracted data from tweeter is stored in CSV file format. Then we conduct the sentiment analysis to get the sentiment of people that initially performs data cleaning, followed by positive, negative or neutral sentiment distributions of tweets with their corresponding sentiment and subjectivity score. This approach uses a Lexicon based approach for sentiment classification and computing subjectivity and polarity score. For our proposed work, we extract a list of tweets using the Cursor() method provide by Tweepy package in Python on the required topics. The subjectivity and polarity score of the corresponding tweet is calculated, based on which they are categorized into their respective sentiment class. Score and these results are visualized using pie chart and trend graph. Tweets are continuously extracted in real time and its output is continuously updated.

II. LITERATURE REVIEW

Shiv Kumar Goel and Sanchita Patil [3] performed sentiment analysis on Twitter data using r programming. They were able to classify the sentiment of tweets by determining their polarity. To achieve this, they made use of the lexicon approach through Word cloud package. Results were presented using pie charts with a respective sentiment score to present results in a better way.

Akash Mahajan, et-al [4] performed an analysis on the impact if government programs for their research. For this, they collected data from an official government site. They processed their data using the Stemmer algorithm. This made their input data more compact to perform sentiment analysis. Data is analyzed and categorize into positive, negative and neutral sentiments and presented visually using pie charts. The final outcome provided us with information on the varying levels of sentiments expressed by users towards various government programs like Swachh Bharat Abhiyan, etc. JinCheon Na, et-al [5] performed sentiment analysis at the clause level in their research work to analyze feedback and information from multiple IMDB movie reviews. The system keeps an account of all the different grammatical words in a sentence in which clauses are classified independently in order to calculate sentiment score for each clause focused on a particular aspect to highlight the most positive and negative. Error rates were common due to the presence of similar words or words with many meanings, grammatical errors, unfinished words and sentences, and incorrectly written clauses. This lead to unreliable results. The algorithm was unable to handle some complex expressions of sentiment in the text due to which major misclassifications were made.

As per Zhao Jianqiangi and Gui Xiaolini's findings [6], various text pre-processing methods can each affect the final output of sentiment classification in their own way. They proposed six different data processing methods that show different sentiment polarity classification results in Twitter. A variety of experiments are conducted on five twitter datasets using different classifiers to verify the efficiency of different data processing methods. The following results showed that the efficiency of classifiers can be negligibly affected by the removing of URLs, special characters and stop words from the tweets. Classification accuracy is also improved by replacing negation and expanding acronyms. Therefore, removal of stop words, numbers, and URLs are helpful to reduce noise but performance remain unaffected. Negation replacement is effective for sentiment analysis. Their studies conclude that the selection of necessary data processing methods and feature models for different classifiers can lead to varying results when classifying sentiment in Twitter data.

Mondher Bouazizi and Tomoaki Otsuki (Ohtsuki) [2] used Parts of Speech tags to extract patterns that characterized the level of sarcasm of tweets. They performed different natural language processing (NLP) tasks Using Apache Open NLP tool. Their studies show good results by the applied selected approach but if the training set was bigger results could be much better. Because all the possible sarcastic patterns were not covered due to the small training set.

Zhang, et-al in [7] performed an investigation on cell phone surveys. This approach proved helpful in comparing accuracy. It is beneficial in a judgment of the product grade and standing in the society [7]. They used three machine-learning algorithms (Naïve Bayes, KNN, and Random Forest) to calculate the opinion accuracy with the RF method showing an acceptable performance. There are a few approaches in reading sentiments and opinions. (Godbole, et-al) analyzed information sentiments and blogs [8]. It splits previous data in the context of their precise venture into classes. First class regards the strategies for routinely growing sentiment lexicon and the second pertains to structures that examine sentiment for complete files.

Balahur & Turchi [9] implemented a model that is similar in performance to that of a NB model of a word which investigates the document polarity by taking into account all the important features of the document. This model can extract words with polarity based on the domain it belongs to. This allows it to create unique domain-dependent word polarity dictionaries for every topic that data is assessed on. This results in better performance rate when applying the machine-learning model in appropriate domain or topic.

III. METHODOLOGY

Various studies are performed for sentiment analysis on textual data that primarily analyze unorganized and unstructured data into an organized and structured manner. They then calculate the subjectivity and polarity of opinion and classify the respective opinion as positive or negative and neutral. In this paper, we proposed a methodology that implements machine-learning algorithms to classify the sentiment polarity of the tweets. The proposed System allows us to process the Twitter data and to carry out the analysis. Figure 1 shows the basic architecture of methodology implemented in this study. Figure 2 shows the block diagram for data collection phase for sentiment analysis where data is extracted from twitter and then sentiment analysis is performed on that data in various phases.

The proposed framework for sentiment analysis helps us to classify sentiment and their corresponding subjectivity. The steps for this process are discussed later in this section. Figure 3 shows the block diagram of the data pre-processing phase that helps us to understand the process in a much better way.

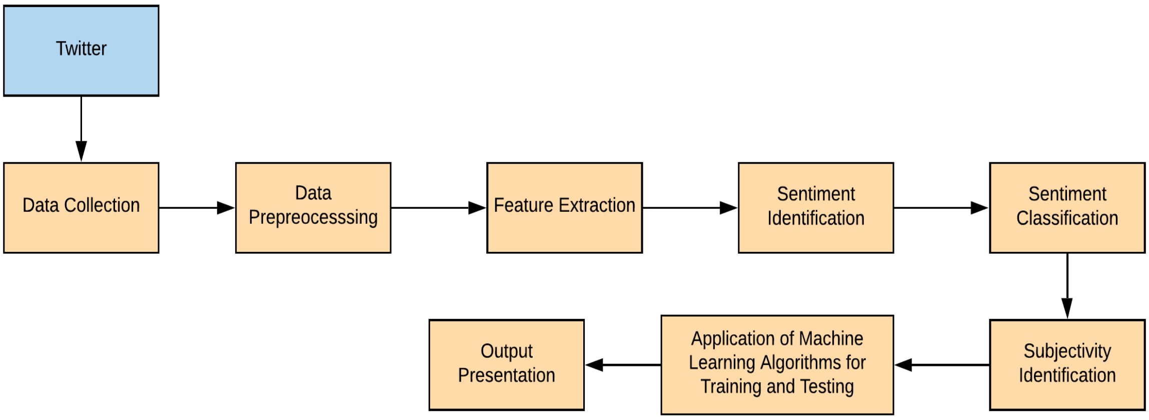

Fig. 1: Basic Architecture of Real-time Sentiment Analysis on Twitter Data

3.1 Data Collection

We chose Twitter for our data collection as it is very popular nowadays. Tweets can be extracted from Twitter using hashtags or keywords. Hashtags or keywords are basically used to categorize tweets, making it easy to search. To extract tweets from twitter, twitter API is required. The Twitter Search API could produce a limited number of tweets at a time which is one of the constraints imposed by Twitter. The Twitter API can be accessed through the Tweepy library in python. Tweepy allows us to easily use the twitter streaming API. It manages authentication, connection, and many other services. API authentication is necessary for accessing Twitter streams. The Tweepy Cursor method can be used to search and download twitter messages in real time from a given date based on a search term. It is useful for obtaining a high volume of tweets. Figure 2 shows the block diagram for data collection.

3.1.1 Authenticating Twitter Access

It is essential for us to obtain the following features from Twitter to access the Twitter Streaming API: API key, API secret, Access token and Access token secret. These can be obtained by creating a Twitter application. It requires a developer's account. If you do not already have a twitter developer's account, then follow the steps below:

1) *Step 1:* Create a Twitter developer account using dev.twitter.com.
2) *Step 2:* Next, go to https://apps.twitter.com/ and sign in with your Twitter credentials.
3) *Step 3:* Click on "Create New App". Fill out the form, agree to the terms, and click "Create your Twitter application".
4) *Step 4:* After creating the app, go to "Keys and tokens" tab, and copy your "API key" and "API secret"
5) *Step 5:* Scroll down and click "Create my access token", and copy your "Access token" and "Access token secret".

In this way, we have all the necessary keys and tokens using which the API can authenticate itself with the Twitter Authentication server.

3.1.2 Accessing Twitter Data

After the authentication, we need to connect with Twitter Streaming API. Tweepy, a python library enables us to connect with Twitter and download data. Once the Twitter Authentication service authenticates the API, a token is generated and made available to the API for each Twitter server transaction. Using this token, tweets are mined using hashtags or keywords. To access the data, we use the Cursor() method to find tweets referring to a specified search term. This is implemented using a search term and start date parameters from which we wish retrieve tweets. We can control the number of tweets extracted by specifying a numerical parameter for the items() function in Cursor() method. 1000 real time tweets on President Trump using #Trump and 1000 real time tweets on Prime Minister Modi using #Modi are collected and stored separately in .csv files as datasets.

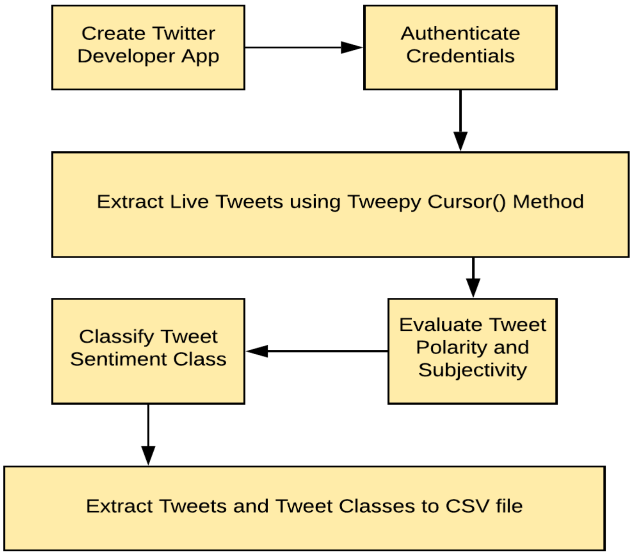
Create Twitter Developer App
Authenticate Credentials
Extract Live Tweets using Tweepy Cursor() Method
Evaluate Tweet Polarity and Subjectivity
Classify Tweet Sentiment Class
Extract Tweets and Tweet Classes to CSV file

Fig. 2: Block Diagram for Data Collection Phase

3.2 Data Preprocessing

The data which is extracted from twitter is dirty and unorganized, expressed in different ways by using various vocabularies, duplicate characters, vulgar language, unique words with no proper meaning, etc. Therefore, data preprocessing focuses on data cleaning and stop word removal. Figure 3 shows the block diagram for the techniques used to identify the features and categories for each class of tweets.

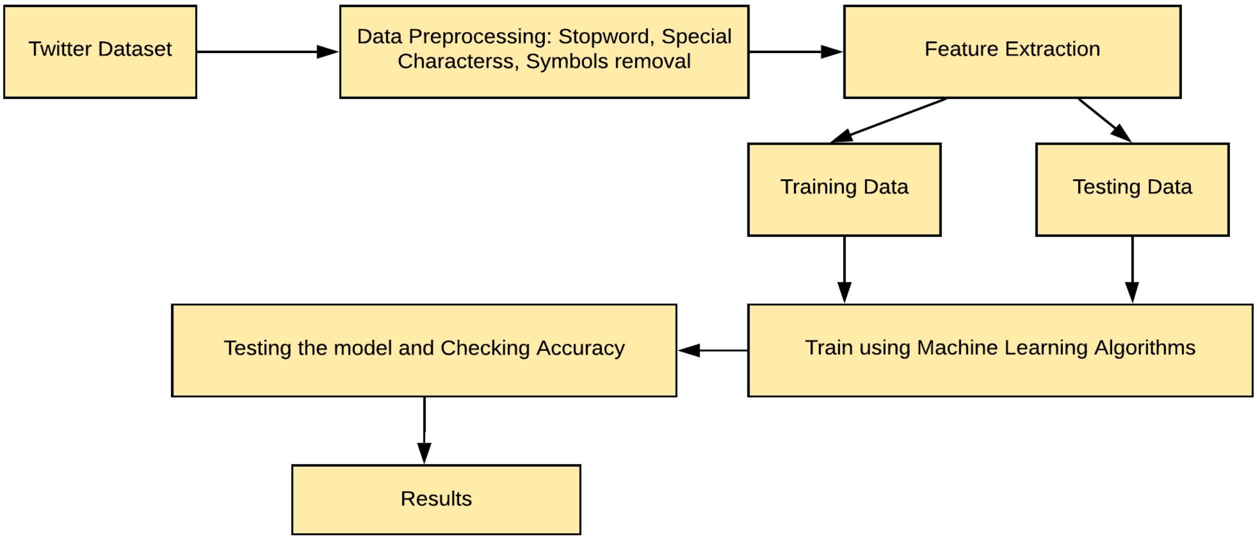
Twitter Dataset
Data Preprocessing: Stopword, Special Charactersss, Symbols removal
Feature Extraction
Training Data
Testing Data
Testing the model and Checking Accuracy
Train using Machine Learning Algorithms
Results

Figure 3: Block Diagram for Data Pre-processing Phase

3.2.1 Data Cleaning

It includes Removal of unnecessary data such as HTML Tags, white Spaces and Special characters takes place from the Twitter dataset. This noise does not make any sense, therefore, they need to be removed. Data cleaning is achieved by importing regular expression (RE) python library. Process of cleaning data for our system is as follows:

1) First, the URLs are removed for simplifying the understanding of the tweets for sentiment classification.
2) Twitter handlers such as '@fwc18' are also removed as they don't provide any information in expressing sentiment.
3) Next, punctuation and special characters are removed.
4) The non-textual and irrelevant contents are identified and removed.
5) Next, extra white spaces are replaced with single white space.
6) Finally, the white spaces from the beginning and end are also removed.

3.2.2 Stop Words Removal

Stop words are the dictionary-based bag of words. These are words that are regularly used in every language. Stop words focus on important word only instead of commonly used words in a language. Stop word removal is done by eliminating the unnecessary words from the Twitter data set. Thus, the resultant data set will only contain the required information for the analysis. The process of stop word removal is as follows:

1) First, tokenization takes place. "Tokens" are usually individual words and "tokenization" is taking a text or set of text and breaking it up into its individual words.
2) After that, all the unnecessary words are removed after tokenization such as 'a', 'an', 'the', and so on. These unnecessary words are the stop words which have no meaning.

After stop word removal, only important words that could lead to sentiment detection are left. Stop word Removal and tokenization is achieved by another python library known as NLTK.

3.2.3 Lemmatization

Lemmatization is the process that is implemented for reducing inflectional forms as well as related forms that are derived from a word to a common or single base form. For instance, vehicle, vehicles, vehicle's, vehicles' are all interpreted as vehicle. Lemmatization is similar to stemming with a difference. Lemmatization involves correctly using a term vocabulary and morphological analysis, normally aimed solely at removing inflection marks and returning to the base or dictionary form of a word recognized as the lemma. Whereas stemming focuses on generating morphological variants of an original word. It is basically the implementation of a coarse heuristic method that chops off the ends of words in the expectation of attaining this objective properly most of the time, and often involves removing derivative affixes. If faced with the token "likely", stemming might only return "l", whereas lemmatization would try to return either "likely" or "likes" based on whether the token was used as a verb or a noun. The words can also vary in the fact that stemming most commonly collapses derivative-related words, whereas lemmatization usually collapses only distinct inflection forms of a lemma. Using the TextBlob python library, lemmatization is performed to conduct easy natural language processing tasks.

3.3 Feature Extraction

Feature extraction is an important step in opinion mining that generates a list of object, aspect, features, and opinions. The purpose of feature extraction is to extract opinion sentences which contain one or more features, aspects, and opinions. In such cases, features are usually nouns, noun phrases, and their tweet opinion usually being expressed through adjectives and adverbs. In this study, features are extracted using the TextBlob library. After preprocessing the data, only necessary words are left in tweets which are then consequently used for analysis. First, the nouns and noun phrases are extracted from the tweets. These noun phrases are used to know the subject in the tweet. After the extraction of nouns and noun phrases, only words that have features or aspects such as adjective and adverb and so on are left. Therefore, in the next phase, these extracted features are classified into sentiments.

3.4 Sentiment Identification

After feature extraction, we identify the positive and negative orientation of words. These features are searched into opinion word list from the huge set of corpora in TextBlob Library to find out the sentiments. Features are searched into positive and negative word list of the dictionary. If the word in the respective tweet or sentence is present in the positive opinion word list, then the feature is assessed as a positive sentiment. If the word in the respective tweet or sentence is present in the negative opinion word list, then the feature is assessed as a negative sentiment. If the word is not present in either list, then the sentiment is considered as neutral. So, the final Polarity Score for the tweet is calculated by subtracting the negative score from the positive score. The polarity score is a float ranges from [-1 to 1]. This polarity score allows us to classify the tweets according to polarity which is discussed in the next phase.

3.5 Sentiment Classification

In this phase, we classify the sentiment of the tweet by using our calculated sentiment polarity score. We classify the sentiments for both President Trump and Narendra Modi. The sentiments are classified into 3 categories as positive, negative and neutral. When assessing polarity score of the tweet, getting score <0 means that the tweet is conveying a negative sentiment, and a score of >0 means that the tweet is expressing a positive sentiment, while a score of 0 implies that the tweet is expressed neutrally. Once the tweets are classified, they are then split into two categories; the tweets that express negative sentiment and the tweets that express non-negative sentiment.

3.6 Subjectivity Identification

Subjectivity can be seen in the explanatory sentences. Subjective sentences are opinions that define the emotions of individuals towards a particular topic or subject. There are many forms of subjective expressions, such as opinions, viewpoints, thoughts, outlooks, perspectives. The subjective phrase is "He has a fondness for meat dishes," although the objective phrase includes facts and has no view or opinion. For instance, “She has a pen that is pink in color” is an objective sentence. A subjective sentence may not express any sentiment. For example, "I prefer a laptop with a good graphics card and processor" is a subjective sentence, but does not express any sentiment. In our system, we get the subjectivity score for the tweets using the TextBlob library function. TextBlob Library already has a dictionary that contains subjectivity score for the words. Modifiers increase the subjectivity of a word or sentence. For example, "Very much" is more subjective than "much". When assessing subjectivity of the tweet, getting a score of 0 implies that the sentence is a fact, and getting a score of 1 implies that the tweet is conveying an opinion or expressing emotion.

3.7 Application of Machine Learning Algorithms

When classifying tweets using machine-learning algorithms, it is usually preferable to implement these algorithms on labeled input data. In this case, we apply classifiers such as NB and Random Forest Classifier on the twitter data.

3.7.1 Naïve Bayes Algorithms

The Bayesian classification model is a probabilistic classifier machine learning technique. It assumes each feature is conditional independent to other features given the class, that is:

$$P(c \mid t) = \frac{P(c)P(t \mid c)}{p(t)} \tag{1}$$

Where “c” is a specific class and “t” is the tweet we wish to classify. P(c) and P(t) are the prior probabilities of class and tweet. And P(t | c) is the probability the tweet appears given this class. In this case, the value of class c might be 0 (Negative) or 1 (Positive or Neutral) and t is a sentence.

The final outcome is to determine which value of “c” can maximize P(c | t)

Where P(wi | c) is the probability of the ith feature in tweet “t” appearing given classification of “c”. It is imperative that we obtain and train parameters of P(c) and P(wi | c). They are calculated by determining Maximum Likelihood Estimation of each other.

When predicting class of new tweet “t”, it is necessary to determine the log likelihood log P(c) + ΣilogP(wi|c) for different classes, and take the class with highest log likelihood as prediction.

3.7.2 Random Forest Classifier

Random forests are an ensemble learning technique for classification that operate by constructing a large number of decision trees during the training process with the final output or class being the category that happens to be the mode of the categories output by discrete trees. The main focus of this process is to generate multi-altitude decision trees during the input phase with the final output being generated in the manner of multiple decision trees. The association between trees can be removed or reduced by indiscriminately choosing trees, leading to an increase in the resulting prediction power as well as the potency of algorithm. The predictions are created based on the aggregation of the predictions of assorted ensemble data sets.

It may be defined as the gathering of tree-dependent classifiers. It performs its function by splitting every node by using the first-rate node amongst randomly selected predictors at that node.

The original data is replaced with newly created data that is used for training. Random characteristic selection is used to grow new trees. These are not pruned.

The random forests algorithm (for both classification and regression) is as follows:

1) Randomly select N records from the input dataset.
2) Make a decision tree based on these N records.
3) Number of trees required for algorithm are specified and the above steps are repeated.
4) Each tree predicts new information through aggregating the predictions of the N trees.
5) Finally, the new record is assigned to the prediction class with majority.

3.8 Performance Evaluation Parameters

These are the parameters that help us evaluate the performance of an algorithm. They are determined based on the elements of a confusion matrix.

From the confusion matrix, terms such as “True Positive (TP)”, “False Positive (FP)”, “True Negative (TN)”, and “False Negative (FP)” are determined and are used to compare label of classes in this matrix.

- “True Positive (TP)”: No. of positive reviews that are correctly classified as positive.
- “False Positive (FP)”: No. of positive reviews that are falsely classified.
- “True Negative (TN)”: No. of negative reviews that are correctly classified as negative.
- “False Negative (FN)”: No. of negative reviews that are falsely classified.

The above terms are used to calculate performance evaluation parameters such as Precision, Recall, F-Score, Accuracy, etc. These values allow us critically evaluate our algorithms as well as the prediction quality and performance.

Precision: It is characterized as the proportion of number of models effectively named as positive to the total addition of number of emphatically arranged positive model.

$$\text{Precision} = \frac{TP}{(T\,P + F\,P)}$$

Recall: It quantifies the culmination of the classifier result. It is the proportion of complete number of positively labeled model to total number of positive models.

$$\text{Recall} = \frac{TP}{(T\,P + F\,N)}$$

F-measure: It is the symphonious mean of precision and recall. It is required to streamline the framework towards either precision or recall, which ends up having more impact on conclusive outcome.

$$\text{F} - \text{Measure} = \frac{(2 * Precision * Recall)}{(Precision + Recall)}$$

Accuracy: It is determined as the proportion of correctly grouped model to total number of models.

$$\text{Accuracy} = \frac{T\,P + T\,N}{(T\,P + T\,N + F\,P + F\,N)}$$

3.9 Output Presentation

The final phase of our sentiment analysis is the visualization of the results. We use bar charts, performance evaluation parameters and tables to view the results. We use a python library Matplotlib to plot bar charts. Its numerical extension to mathematics is NumPy. And we're using Pandas for table visualization. Pandas is a software library for manipulation and analysis of data written for the Python programming language. It provides data structures and operations to manipulate numerical tables and time sequences. We use the de facto machine-learning library Scikit-learn for dividing the testing and training set for our data. We also use it to import all the algorithms that we wish to implement for this experiment, as well as all the performance parameters that are necessary for evaluating each model in this project.

IV. RESULTS AND DISCUSSION

Data is collected from twitter for analyzing tweets on any popular topic using hashtag or keyword. In our experimental work, we store tweets in two datasets. We choose two trending hashtags #Trump to refer to President Donald Trump and #Modi referring to Indian Prime Minister Narendra Modi. Dataset1 contains the tweets on #Trump and dataset2 contains the tweets on #Modi. These tweets are extracted from Twitter with the start date specified to be 5 August 2020. #Trump is the hashtag for US President Donald Trump as he is trending around the world due to the upcoming Presidential elections in the United States. The hashtag #Modi refers to Indian Prime Minister Narendra Modi who is trending due to his new policies in response to the Corona Virus pandemic. Dataset1 and Dataset2 both contain 1000 tweets. Figure 1 shows the bar chart for categorization of tweet classification for 1000 tweets on #Trump in Dataset1 and for 1000 tweets on #Modi in Dataset2.

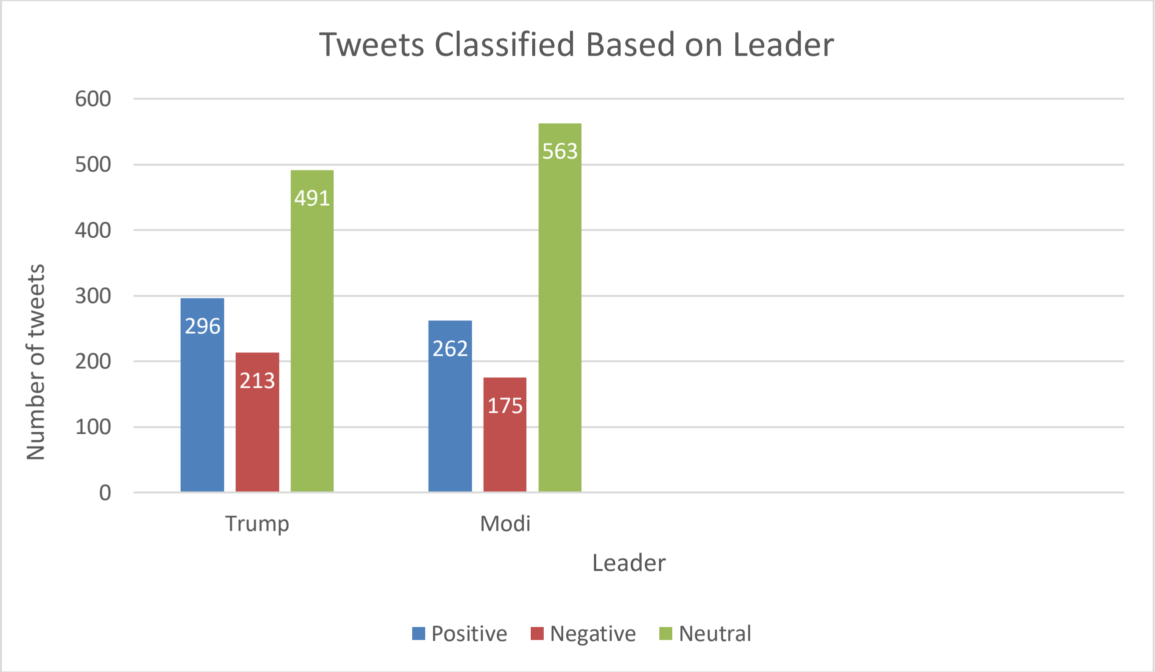

Figure 4: Bar Graph of Tweets Classified Based on Leader

According to Bar Graph, we infer that out of 1000 tweets on #Trump collected in real time: 296 (29.6%) tweets are positive, 213 (21.3%) tweets are negative, and the rest 491 (49.1%) tweets are neutral. These results show that the majority of people share a neutral opinion or view towards #Trump. Similarly, we can infer that out of 1000 tweets on #Modi collected in real time: 175 (17.5%) tweets are Negative, 563 (56.3%) tweets are neutral and rest 262 (26.2%) tweets are positive. These results show that the majority of people also carry neutral sentiment towards Modi.

The next stage is to implement the machine-learning algorithms on both datasets. We then evaluate the performance parameters for tweets that were classified as negative (value 0) and non-negative (value 1), i.e, positive or neutral sentiment. The model reports of the algorithms implemented for 1000 tweets on #Trump and #Modi each are summarized in a tabular format in Table 4.1.

Table 4.1: Comparison Table of Algorithm Performance

Method	No. of Tweets	Leader	Value	Precision	Recall	F-Score	Accuracy
Multinomial Naïve Bayes	1000	Trump	0	0.92	0.29	0.44	86%
			1	0.86	0.99	0.92	
Random Forest Classifier (n=200)	1000	Trump	0	0.75	0.24	0.36	84%
			1	0.85	0.98	0.91	
Multinomial Naïve Bayes	1000	Modi	0	0.86	0.35	0.50	88%
			1	0.88	0.99	0.93	
Random Forest Classifier (n=200)	1000	Modi	0	0.91	0.29	0.44	87.5%
			1	0.87	0.99	0.93	

According to the Table, the Random Forest Classifier achieved 84% accuracy when predicting negative (0) and non-negative (1) sentiment class for tweets on #Trump and it also achieved 87.5% accuracy when predicting negative (0) and non-negative (1) sentiment class for tweets on #Modi. The F1 Score is needed for achieving balance between Precision and Recall. This Model has an overall high precision rate, but a low recall and F1 score when predicting negative tweets. It high recall and F1 score when predicting non-negative (positive or neutral) tweets due to larger portion of the dataset having positive or neutral sentiment. Having a lesser recall rate indicates the presence of many false negatives.

According to the Table, the Multinomial Naïve Bayes Model achieved 86% accuracy when predicting negative (0) and non-negative (1) sentiment class for tweets on #Trump and it also achieved 88% accuracy when predicting negative (0) and non-negative (1) sentiment class for tweets on #Modi. This Model has an overall high precision rate, but a low recall and F1 score when predicting negative tweets. It has high recall and F1 score when predicting non-negative (positive or neutral) tweets due to larger portion of the input dataset having positive or neutral sentiment. Having a lesser recall rate indicates the presence of many false negatives.

We can infer from the table above that both algorithms have greater accuracy when implemented on a larger dataset with lesser number of features.

Both algorithms have an acceptable accuracy though Random Forest can prove more useful for a much larger dataset due to its unbiased approach of predicting the sentiment class of tweet by relying on majority value of class. As this project focuses on text classification, it is the perfect algorithm to run for this study in spite of its longer runtime.

Despite achieving an acceptable accuracy for this data, the concept of independence assumption in Naïve Bayes can lead to problems in classification, especially in the case of tweets with negative words preceding adjectives. Despite its relatively high accuracy and simplicity, it might prove to be unreliable with lack of proper features or input data.

V. CONCLUSION

In this work, a methodology is proposed using machine-learning algorithms to perform sentiment analysis on two datasets of real time tweets extracted from Twitter. Dataset1 contains 1000 tweets on #Trump and classified results are visualized using a detailed bar chart and classification report. The overall nature of dataset1 is mostly neutral due to larger number of neutral tweets and more positive tweets than negative ones. While the dataset2 contains 1000 tweets on #Modi and classified results are visualized using bar chart and classification report. The overall nature of dataset2 is also mostly neutral with larger number of neutral tweets and more positive tweets than negative.

As Twitter data continuously gets updated due to many people posting on the same time on the same topic, data changes and varies the more we run this program. Thus the accuracy of classifiers constantly varies the more data is received as well as being time consuming.

From the results, it was shown that both Random Forest and Naïve Bayes algorithms had acceptable accuracy (>84%) when predicting negative and non-negative (positive or neutral) sentiment class of the tweets. Despite this, they both had but low recall and f1-score when predicting negative tweets, due to large imbalance between number of negative and non-negative tweets.

.

REFERENCES

[1] Mark E. Larsen, Tjeerd W. Boonstra, Philip J. Batterham, Bridianne O'Dea, Cecile Paris, Helen Christensen, "We Feel: Mapping Emotion on Twitter", IEEE Journal Of Biomedical And Health Informatics, Vol. 19, NO. 4, JULY 2015.

[2] Mondher Bouazizi, Tomoaki Otsuki (Ohtsuki), "Pattern- Based Approach for Sarcasm Detection on Twitter", IEEE Access, volume 4, 2016.

[3] Shiv Kumar Goel, Sanchita Patil, "Twitter Sentiment Analysis of Demonetization on Citizens of INDIA using R", IJIRCCE, Vol. 5, Issue 5, May2017.

[4] Akash Mahajan, Rushikesh Divyavir, Nishant Kumar, Chetan Gade, L.A. Deshpande, "Analyzing the Impact of Government Programs", IJIRCCE, Vol. 4, Issue 3, March 2016.

[5] Tun Thura Thet, Jin-Cheon Na, Christopher S.G. Khoo, "Aspect-based sentiment analysis of movie reviews on discussion boards", Journal of Information Science, 2010, pp. 823–848

[6] Zhao Jianqiangi, Gui Xiaolini, "Comparison Research on Text Pre-processing Methods on Twitter Sentiment Analysis", IEEE. Translations and content mining, 2017

[7] Zhang, L., Hua, K., Wang, H., and Qian, G.,"Sentiments reviews for mobile devices products", The 11th International Conference on Mobile Systems and Pervasive Computing (MobiSPC-2014) ,procedia computer scinence, Volume 34, 2014.

[8] Godbole, N., Srinivasaiah, M., and Skiena, S., "Large-Scale Sentiment Analysis for News and Blogs", ICWSM'2007 Boulder, Colorado, USA, 2007.

[9] Balahur, A. & Turchi, M., "Multilingual Sentiment Analysis using Machine Translation? ", In Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis. Korea, 2012.