://www.overleaf.com/project/62e6c3b4c4dc504f2310f98b

Sentiment Analysis using Machine Learning and Recurrent Neural Networks

Chayanika Basak
Department of Artificial Intelligence and Data Science
Indira Gandhi Delhi Technical University for Women
Delhi, India
chayanika071btcsai21@igdtuw.ac.in

Aaliyah Beg
Department of Computer Science
Jamia Millia Islamia
Delhi, India
aaliyahbeg123@gmail.com

Pooja Kumari
Department of Artificial Intelligence and Data Science
Indira Gandhi Delhi Technical University for Women
Delhi, India
pooja051btcsai21@igdtuw.ac.in

Ritu Rani
Department of Artificial Intelligence and Data Science
Indira Gandhi Delhi Technical University for Women
Delhi, India
riturani@igdtuw.ac.in

Arun Sharma
Centre of excellence
Indira Gandhi Delhi Technical University for Women
Delhi, India
arunsharma@igdtuw.ac.in

Amita Dev
Centre of excellence
Indira Gandhi Delhi Technical University for Women
Delhi, India
vc@igdtuw.ac.in

***Abstract*—Sentiment Analysis is an NLP problem dealing with the understanding of emotions and assigning the tag of negative or positive to the tweets. For this purpose we have used carefully picked classification models inclusive of Bernoulli NB which yields accuracy 81.39%, Multinomial NB which yields accuracy 81.72%, Support vector Machines which yields accuracy 88.41% and Logistic Regression with best accuracy 80.62% among the ML models and the DL model Recurrent Neural Networks which gave accuracy 81.45% and through in-depth analysis of our data set we have preprocessed it in such a way that the dataset is clean and set to be trained in the models without underfitting or overfitting.**

***Index Terms*—sentiment, twitter, bernoulli naive bayes, multinomial naive bayes, logistic regression, support vector machine, recurrent neural network, natural language processing**

I. Introduction

Microblogging services like Twitter have seen tremendous growth in use over the last few years. As a result of this expansion, businesses and media outlets are looking for more and more methods to tap into Twitter to learn what consumers think and feel about their goods and services. There has been some research on how emotions are expressed in domains like consumer reviews and news articles, but there has been far less research on how emotions are transmitted in informal languages and short messages with length constraints.

This project aims to develop an algorithm that accurately classifies tweets about a search query as positive or negative. We believe that we can accurately categorize emotions in Twitter messages by using machine learning techniques. This kind of sentiment analysis is typically helpful for consumers looking to explore a product or service or for marketers looking to understand how the public feels about their business.

II. Motivation

Sentiment Analysis is a big step towards training machines and robots to understand human emotions. It not only gives hope that emotions can be inculcated into inanimate objects but also opens doors to a wide arena of opportunities for predictions and other crucial tasks. This project was a step for us to research on the same by taking into account a microblogging service such as twitter, which has a large collection of data for analysis in multiple languages by people of multiple cultures. The data from twitter provides us with enough variety to make our model more general than specific and hence has been used in our study. We hope to achieve results that help us accurately predict human emotions through machine learning and deep learning models.

III. Literature Survey

In a study, Saleena et.al [1], gave an ensemble classification system for twitter sentiment analysis was discussed and the results indicated that it outperformed both the majority voting ensemble classifier and stand-alone classifiers.

In another study Ahmad et.al [2], discussed sentiment analysis of tweets using SVM. The outcomes clearly demonstrated how SVM performance is dependent on the input dataset and the importance of vast and varied datasets for better performance.

Pratama, Yohanssen and others implemented sentiment analysis on twitter using Naïve Bayes algorithm to know the people responses to debate of DKI Jakarta governor election [3] and achieved an accuracy of 96%.

Another similar study done by Rachman et.al [4] used tf-idf and logistic regression on tweets related to Covid-19 collected

on 30 April 2020 and made predictions with an accuracy of 94.71%.

Some experiments conducted by Jianqiang et.al [5] reveal that expanding acronyms and replacing negation enhanced the performance of the classifiers, but deleting URLs, removing numerals, or removing stop words didn't.

According to Praveen et.al study [6], the Naive Bayes algorithm and NLTK perform rather well for negative comments but not with tweets having ironic, snarky, or contain problematic context or references.

The combination of an ensemble cluster with an SVM classifier, as carried out by the C^3E-SL method, can improve tweet classification, according to experimental results given in a work by Coletta et.al [7] on four datasets. Another study done by Wagh et.al [8] demonstrates that accuracy of sentiment analysis increases when SVM, Naive-Bayes, and maximum entropy are used after semantic analysis WordNet. Additionally, a hybrid technique can increase accuracy by up to 4%.

A research involving stock predictions using twitter sentiment analysis proposed by Mittal et.al [9] found that people's mood affect their investment decisions.

Two methods for classifying Arabic text were compared in a study successfully carried by Elghazaly et.al [10]. The Naive Bayes technique is a well-liked one for this application because it is quick and quite accurate, according to the results.

The experimental results in Dey et.al paper [11] demonstrated that the Naive Bayes model surpassed the k-NN approach and achieved accuracy rates of above 80%.

In the article published by Raschka et.al [12] text classification using naive bayes has been discussed in detail along with all the pre-processing techniques.

IV. Dataset

It is present on the Kaggle website under the name "Sentiment140 dataset with 1.6 million tweets". It has 1600000 rows and 6 columns. This data has been cleaned by dropping unnecessary columns that have no impact on our output. For example, the 'Flag' column was dropped as it had the same value for all records. The 'Date' column was split into 'Day', 'Month', 'Year' and 'Time' for better accuracy of models by targeting very specific features.

Further, it was found that there are only two distinct labels 0 and 4 in the target column and hence those were converted to 0 and 1 for binary classification. Target=0 represents negative(sad,upset,angry etc.) tweets while target=1 represents positive(happy,excited,glad etc.) tweets.

V. Methodology

A. Stop Words

Stop words are terms that are frequently seen in texts and are therefore viewed as being somewhat meaningless. For example, words such as a, an, the, so, and, or etc. Searching against a stop word dictionary that is particular to a given language is one method for removing stop words. We have applied this approach on our data by manually creating a dictionary of stop words relevant to our study and using it to strip the tweets off of all irrelevant words.

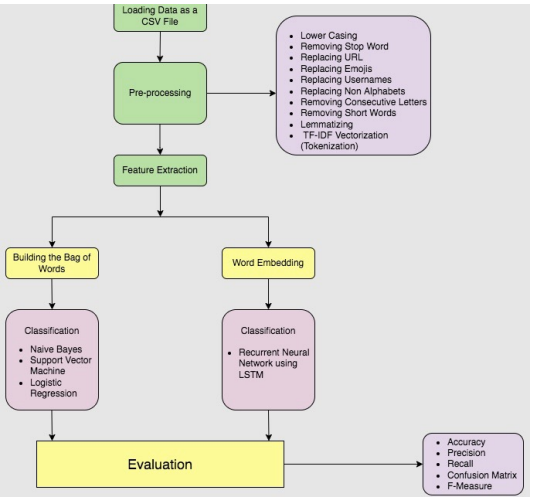

Fig. 1. Methodology Flow

B. Preprocessing on the Dataset

Each tweet has first been converted into lowercase. All the urls in the tweets have then been replaced with the word "URL". Emojis have been replaced with the word "EMOJI" and a value from a emoji dictionary having keys as emojis and values as meanings of the emojis. Further, all usernames have been replaced with the word "USER" in order to hide personal information. All non-alphabetical characters have also been removed. Lastly words having more than 2 consecutive letters have been shortened and short words having a length less than 2 have been ignored.

C. Stemming and Lemmatization

Stemming is the transformation of a term into its root or fundamental form. This process of stemming can result in words that are not real.

Lemmatization, as opposed to stemming, tries to produce the grammatically correct forms of the words, or lemmas. In practise, lemmatization and stemming have similar effects on the effectiveness of text categorization, despite lemmatization being computationally more challenging and expensive than stemming.

After preprocessing, lemmatization has been done on the data to avoid generation of unnecessary and meaningless words.

D. Term Frequency - Inverse Document Frequency

By using the statistical method TF-IDF (term frequency-inverse document frequency), one can determine the significance of a word in each document in a group of documents. This is the next stage of preprocessing in our study.

To accomplish this, the frequency of a word within a document and its inverse document frequency among a collection of documents are multiplied.

It has various applications, with automated text analysis being the most crucial, and is especially helpful for word scoring in ML algorithms for NLP.

TF-IDF was created for searching documents and retrieving information. It works by escalating in accordance with how frequently a term occurs in a document, but is balanced by how many doxuments contain the word. Because they aren't very crucial to any document, terms such as "this", "so", "and" and "or" are ranked low even if they may be used frequently.

Term Frequency: The simplest way to calculate this frequency is to simply count the instances of each word in the text. Other ways to change frequency include the document's length or the frequency of the term that occurs most often.

$$normalized - term - frequency = \frac{tf(term, doc)}{n_{doc}} \quad (1)$$

where,
term = a certain word in the document
doc = the document
n_doc = number of terms in the document [12]

Inverse Document Frequency: This refers to how prevalent or uncommon a word is throughout all documents. A word is more common the nearer it is to 0. It can be calculated by taking the logarithm of the number of documents, divided by the total number of documents containing a term. Therefore, if the word is frequently used and occurs in several documents, this value will be very close to zero. If not, it will be quite close to 1.

$$idf(p) = \log \frac{n_{doc}}{n_{doc}(p)} \quad (2)$$

where,
n_d = Total no. of documents.
$n_d(t)$ = No. of documents containing the term p. [12]

The result of multiplying these two figures is the word's TF-IDF score in a document. The more significant a word is in a given document, the higher the score.

$$tf - idf = tf_n(p, doc).idf(p) \quad (3)$$

The tf-idf vectorizer takes into account the concept of tokenixation and n-grams that have been discussed in sections V-E and V-F;

E. Tokenization

Tokenization is the broad term for the process of dissecting a text corpus into discrete components for use as input by different natural language processing algorithms. Tokens can be thought of as components, such as a word in a sentence or a sentence in a paragraph. Tokenization can be done before or after cleaning the data. We have performed tokenization on our data after the preprocessing stage. Tokenization has been indirectly when the training data was fed into a tf-idf vectorizer.

F. N-grams

A sequence of n elements can be thought of as a token in the n-gram model. For instance, unigram(1-gram) is said to be used when a sentence is broken down to a sequence of single words. The best value for n depends on the language and the application in question. In this study, both unigram and bigram models have been used in order to increase the efficacy of the models.

G. Bag of Words

Prior to using machine learning techniques by fitting the models for training, feature vectors need to be generated for the text. NLP typically uses the "Bag of Words" model. The process of building a vocabulary begins with compiling a list of every word that appears in the training set and assigning a count to each word's occurrences. This vocabulary can be thought of as a sequence of unique items whose order is unimportant.

H. Word Embedding

A type of word representation known as word embeddings enables the representation of words with similar meanings. One of the key developments that may be responsible for deep learning approaches' outstanding performance on challenging natural language processing problems is its distributed representation for text.

*I. **Naive Bayes***

Naive Bayes (NB): It is a classification algorithm based on the Bayes Theorem. It is a classification technique that uses supervised learning. The Bayes Theorem provides us with the likelihood that event A will occur given that event B has previously occurred. In this theorem, A and B are taken to be strongly independent events, i.e., neither affects the other. Additionally, Naive Bayes gives each feature the same weight. It does not favour certain characteristics over the others.

a) Posterior Probability: By updating the prior probability using the Bayes Theorem and taking new information into account, the posterior probability is determined. For example, posterior probability can be used to determine the likelihood that a specific tweet will have positive or negative emotions given the observed feature values in a classification problem like ours. This is illustrated in equation. 4.

$$P(Positive \mid x_i) = \frac{P(x_i \mid Positive) \times P(Positive)}{P(x_i)} \quad (4)$$

where
x_i = feature vector of sample i $\in$ { 1,2,...,n}
$P(Positive \mid x_i)$ = posterior probability that customer will churn, given the set x_i of feature vectors
$P(x_i \mid Positive)$ = Likelihood of a tweet having a set of features x_i given that the tweet is positive
$P(Positive)$ = prior probability that the tweet is positive
$P(x_i)$ = evidence

Given the training data, the main objective of the NB probability is to take the posterior probability and maximise it in order to produce the decision rule as shown in 5.

$$y = argmax_y \left(P(y) \prod_{i=1}^{n} P(x_i \mid y) \right) \tag{5}$$

where
y = class with maximum probability
$P(x_i \mid y)$ = Likelihood of a tweet having a certain set of features x_i given that they belong to class y
$P(y)$ = prior probability of class 'y' [12]

We have just two classes in our data, so ' y ' can take only two values corresponding to 'Positive' and 'Negative'.

Predictions made using the Naive Bayes model have the structure as shown in 6

the tweet is positive if

$$P(Positive \mid x_i) \geq P(Negative \mid x_i) \tag{6}$$

else the tweet is negative

There are three types of Naive Bayes Classifiers namely Multinomial, Gaussian, and Bernoulli Naive Bayes. We have implemented Bernoulli Naive Bayes on our dataset as all the features are binary in nature.

b) Bernoulli Naive Bayes (BNB): It is used when the features of the dataset are binary as it only takes binary values. It is used for discrete data unlike in the case of Gaussian Naive Bayes which is used for continuous data. There can be two mutually exclusive outcomes of a Bernoulli Distribution:
$P(X = 1) = p$
$P(X = 0) = 1 - p.$
where p is the probability of success.

The decision rule for BNB corresponding to our dataset is as shown in the equation. 7

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i) \tag{7}$$

where
x_i = feature vector of sample i $\in$ { 1,2,..,n }

c) Multinomial Naive Bayes (MNB): Bernoulli NB is interested in counts for a single feature that occurs and counts for the same feature that do not occur, whereas Multinomial NB is interested in counts for numerous features that do occur.

In other words, although Bernoulli NB can only focus on a single term, it will also count how many times that keyword appears in the content. For instance, Multinomial NB will categorise a document based on the counts it discovers of various keywords.

As a result, they do model a few different things. Multinomial NB is used when there are discrete many characteristics to consider, which is an option that can be considered for working on our data.

J. Support Vector Machine

Support vector machines (SVMs) are set of supervised ML models based on the concept of a hyperplane, that can be used as classification problem solver to solve a classical two-group problem by categorising new text based on maximum margin hyperplane and achieve good results for text classification tasks like sentiment analysis after providing an SVM model labelled dataset.

- Hyperplane: Our data includes two features. x, y, and Given a pair of (x,y) coordinates, we want a classifier that produces either a positive or a negative output. We place our pre-labeled training data on a plane, and then a SVM uses these data points to produce the hyperplane—which is just a line in 2D plane/line—that best distinguishes the tags. The decision boundary is represented by this line; everything falling on one side of it will be categorised as positive information, and anything falling on the other as negative information.
- Types of SVMs based on the distribution of data in space: Linear case and Non-Linear case
  1) Linear Case: If the data is linearly separable — we could separate the two classes by drawing a straight line and it would fall under the linear case of SVM.
  2) Non Linear Case:In order to deal with nonlinearity, we add a new z-dimension and mandate that it be calculated in a manner that is suitable for us. When dealing with nonlinear data, Support Vector Machine transforms it into a higher dimension where it may be linearly separated. Support Vector Machine does this by utilising various Kernel settings.

K. Logistic Regression

We use this model to predict the probability of a binary event occurring,(yes/no). In place of giving the exact value as 0 or 1, this model gives the probabilistic values between 0 and 1.

- Sigmoid function: Sigmoid function takes any real number and converts it into another value that lies between 0 and 1.

$$S(z) = \frac{1}{1 + (e)^{-z}} \tag{8}$$

  S(z) = Gives output value between 0 and 1, 0 means negative tweets and 1 means positive tweets and 0.5 implies neutral tweet.
  z = input to the function
  e = base of the natural log,
- In logistic regression we used formula for hypothesis i.e.

$$Z = \beta_0 + \beta_1 X \tag{9}$$

$$(h\theta(X)) = \frac{1}{1 + (e)^{-\beta_0 + \beta_1 X}} \tag{10}$$

  where,
  X : weights of the input in our twitter dataset.
  β_0 : logistic regression coefficient

β_1 : logistic regression coefficient associated with X
$h\theta(X)$: Output between 0 and 1

- Cost Function: Cost function J(theta) represents the optimization goal, which is to create a cost function and minimise it in order to create a reliable model with the least amount of error.

$$\mathrm{J}(\theta) = \frac{-1}{m} \sum_{.} [(y)^i \log(h_\theta((x)^i)) + (1 - (y)^i) \log(1 - h\theta(x(i)))] \quad (11)$$

where,
$J(\theta)$ = Accuracy of our prediction model
The more the value of $J(\theta)$, the lesser will be the accuracy and vice versa.
m = number of observations
$h\theta(x(i))$= Output of sigmoid function

- Gradient Descent: By using the gradient descent,we can reduce the cost value. In a recursive process known as gradient descent, the model steadily gets closer to a minimal value. At this point, when the error is minimised and the cost function is optimised, convergence takes place.

L. Recurrent Neural Networks

Due to their internal memory, recurrent neural networks (RNNs) are the most sophisticated method for sequential data. For machine learning issues involving sequential data, it is the first algorithm to recall its input.

- Layer of Embedding- The embedding layer is one of Keras' layers. Although it can be used for various neural network-based tasks, this is mostly used in NLP-related applications, such as language modelling. To resolve NLP difficulties, we can employ pre-trained word embeddings like GloVe. As an alternative, we can train our own embeddings using Keras' embedding layer.
- Long short term memory (LSTM) layer networks were introduced by Hochreiter and Schmiduber. These are widely applied to speech recognition, language modelling, sentiment analysis, and text prediction.

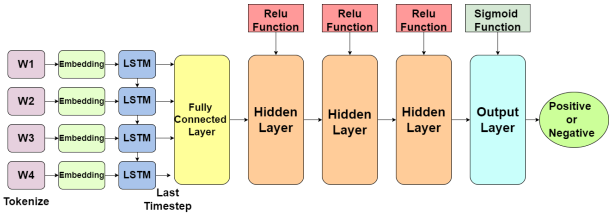

Fig. 2. RNN

- Hidden Layers: RNN's Foundation Recursively calculating hidden states are the capabilities of recurrent neural networks. The hidden state of an RNN can store historical information about the sequence up to the current time step. As a result, this method improves accuracy. We added three hidden layers to our RNN model for the same reason.
- The rectified linear activation function, or ReLU for short, will directly output the input if the input is positive; if the input is negative, it will output zero. It has become the typical activation function for many different kinds of neural networks since a model that uses it is easier to train and commonly outperforms others.
- Sigmoid Function as an Activation Function- The sigmoid function is used since the output of such models is a probability prediction, they are particularly used in this context. Due to the fact that anything only has a probability of occurring between 0 and 1, the sigmoid is the best choice. The function might take numerous forms. As a result, we can calculate the slope of the sigmoid curve between any two points.

VI. Result

Among the machine learning models, Logistic Regression has the highest testing accuracy of 80.62% as observed from table II. The Bernoulli Naive Bayes performs poorly both with training and testing data having a testing accuracy of only 78.25%. Thus, it may be concluded that the Bernoulli Naive Bayes is not a good choice for sentiment analysis here. Multinomial Naive Bayes is, in fact, a better choice as it has slightly higher accuracy (78.34%) and f1-score (0.7834). The Support Vector Classifier performs very well with the training data with 88.41% accuracy but the accuracy drops significantly when it comes to the testing data. This means that the data might be overfitted.

Close observation of the confusion matrices shows that the Logistic Regression model has the lowest percentage of "False Positives" which in our study signifies that it wrongly classifies only 10.61% tweets which have negative emotions as positive tweets. Although in our case, such wrong predictions do not have dire consequences, this property can be quite crucial in some fields of work like disease prediction or in customer attrition prediction models.

ALthough MNB has higher accuracy than BNB, BNB makes higher number of correct predictions when looking at the confusion matrices.

Logistic Regression has the highest precision II which suggests that out of all the negative tweets, the number of correct classifications is highest among all the machine learning models. It also has the highest recall. This signifies that this model has been able to recall the highest negative tweets. As a result, it has the highest f1 score too.

All these results have been obtained after 10 fold cross validation and threshold tuning. The 'Threshold' column in table II shows that all the optimal thresholds obtained after tuning are close to 0.5, which signifies that our data is balanced (almost equal no. of positive and negative tweets).

The ROC curve of all the ML models are quite similar and have very close AUC values which shows that these three models have similar performance. Next a recurrent neural network model was tested. It uses three hidden layers with ReLU activation function and the sigmoid activation function

TABLE I
ACCURACIES OF ALL THE MODELS

Model	Training Accuracy(%)	Testing Accuracy(%)
Bernoulli NB	81.95	78.25
Multinomial NB	82.44	78.34
SVC	88.41	78.25
Logistic Regression	84.80	79.63
RNN	**81.46**	**81.45**

TABLE II
PERFORMANCE OF ALL THE ML MODELS

Model	Precision	Recall	f1-Score	Threshold
Bernoulli NB	0.7838	0.7834	0.7833	0.479
Multinomial NB	0.7837	0.7835	0.7834	0.496
SVC	0.7828	0.7825	0.7825	0.500
Logistic Regression	**0.7965**	**0.7964**	**0.7963**	0.513

in the output layer. After running 3 epochs its accuracy kept increasing from 80.83% to 81.45%.

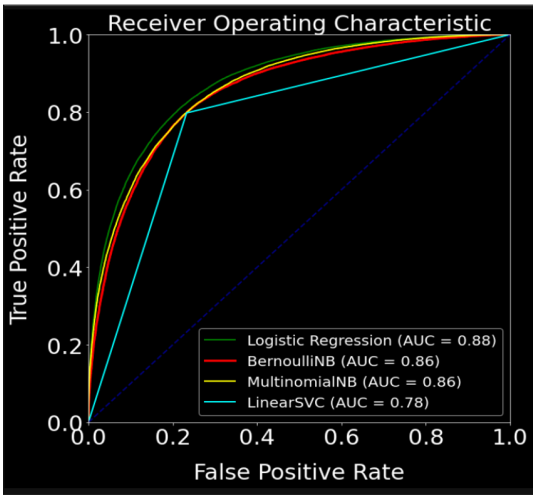

Fig. 3. ROC Curve of all ML models

VII. CONCLUSION AND FUTURE SCOPE

This research produced some fascinating results and insights regarding regarding analysing sentiments using tweets and the various prediction methods available.We have analysed multiple classifiers in order to accurately predict whether a tweet is positive or negative. Our study found that among machine learning classification models, logistic regression had the highest accuracy(80%), precision and recall. Using a Recurrent Neural Network with 3 hidden layers and 3 epochs generated the highest accuracy of 81.45%. In conclusion, neural networks are best suited for sentiment analysis. For even higher accuracy a model with more hidden layers and epochs should be used. Further hyperparameter tuning can greatly increase the accuracies of the classification models. Further study is required to develop the dataset into one with more variety of emotions along with intensities of emotions for targeting specific clients and issues with greater accuracy.

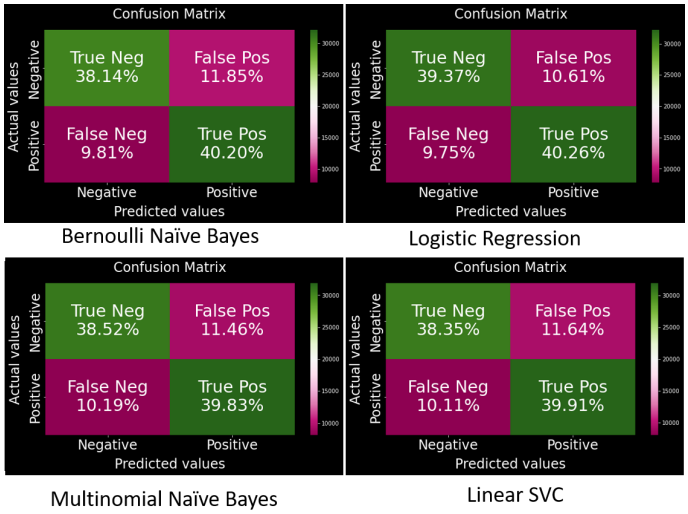

Fig. 4. Confusion Matrices

REFERENCES

[1] Saleena, Nabizath. "An ensemble classification system for twitter sentiment analysis." Procedia computer science 132 (2018): 937-946.

[2] Ahmad, Munir, Shabib Aftab, and Iftikhar Ali. "Sentiment analysis of tweets using svm." Int. J. Comput. Appl 177.5 (2017): 25-29.

[3] Pratama, Yohanssen, et al. "Implementation of sentiment analysis on Twitter using Naïve Bayes algorithm to know the people responses to the debate of DKI Jakarta governor election." Journal of Physics: Conference Series. Vol. 1175. No. 1. IOP Publishing, 2019.

[4] Rachman, Fika Hastarita. "Twitter sentiment analysis of Covid-19 using term weighting TF-IDF and logistic regresion." 2020 6th Information Technology International Seminar (ITIS). IEEE, 2020.

[5] Jianqiang, Zhao, and Gui Xiaolin. "Comparison research on text preprocessing methods on twitter sentiment analysis." IEEE access 5 (2017): 2870-2879.

[6] Parveen, Huma, and Shikha Pandey. "Sentiment analysis on Twitter Data-set using Naive Bayes algorithm." 2016 2nd international conference on applied and theoretical computing and communication technology (iCATccT). IEEE, 2016.

[7] Coletta, Luiz FS, et al. "Combining classification and clustering for tweet sentiment analysis." 2014 Brazilian conference on intelligent systems. IEEE, 2014.

[8] Wagh, Rasika, and Payal Punde. "Survey on sentiment analysis using twitter dataset." 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2018.

[9] Mittal, Anshul, and Arpit Goel. "Stock prediction using twitter sentiment analysis." Standford University, CS229 (2011 http://cs229. stanford. edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis. pdf) 15 (2012): 2352.

[10] Elghazaly, Tarek, Amal Mahmoud, and Hesham A. Hefny. "Political sentiment analysis using twitter data." Proceedings of the International Conference on Internet of things and Cloud Computing. 2016.

[11] Dey, Lopamudra, et al. "Sentiment analysis of review datasets using naive bayes and k-nn classifier." arXiv preprint arXiv:1610.09982 (2016).

[12] Raschka, Sebastian. "Naive bayes and text classification i-introduction and theory." arXiv preprint arXiv:1410.5329 (2014).