Comparative Analysis of Different Machine Learning Algorithms for Twitter Sentiment Analysis

Ashutosh Tiwari
Department of Information Technology
Terna Engineering College, Nerul
Navi Mumbai, India
ashu.atiwa@gmail.com

Shreyas Chikhalikar
Department of Information Technology
Terna Engineering College, Nerul
Navi Mumbai, India
shreyaschikhalikar@gmail.com

Anup Singh
Department of Information Technology
Terna Engineering College, Nerul
Navi Mumbai, India
anupsingh00748@gmail.com

Dakshata Argade
Department of Information Technology
Terna Engineering College, Nerul
Navi Mumbai, India
dkshtargade@gmail.com

***Abstract*-- Twitter claims that 53% of its users are likely to try new products. In such a scenario, Twitter can be a good medium to advertise a new product. The success of such a product can be determined by the reviews given by Twitter users. As a result, Twitter can be influential in determining the user's opinion about the product. Machine Learning (ML) can be used to perform Twitter Sentiment Analysis (TSA) to extract meaningful information from the tweets. The objective of the project is to determine which Machine Learning algorithm performs better on Twitter data for Sentiment Analysis (SA). This is done by a technique called Natural Language Processing (NLP). The ML algorithm implemented are Logistic Regression (LR) and Naïve Bayes (NB). The training and validation testing are performed on a dataset obtained from Natural Language Toolkit (NLTK). The final testing is performed on a completely different dataset to eradicate any bias originating from the training dataset. The efficiency of both the algorithms are compared and Naïve Bayes outperformed Logistic Regression in terms of efficiency. This result is in accordance with the result of research previously conducted on the same topic.**

***Keywords*-- Twitter Sentiment Analysis, Machine Learning, Natural Language Processing, Natural Language Toolkit, Logistic Regression, Naïve Bayes**

I. Introduction

Twitter is a microblogging site based in San Francisco which serves monthly active users up to 330 million and daily active users of about 192 million as of the year 2020. The user base of the 'SMS of Internet' i.e. Twitter is vastly diverse with users from all countries. More than 500 million tweets are sent each day (Mention, 2018) which shows that Twitter users are very engaging on the social networking platform. The large and distributed network of users is evident that marketers use this platform to showcase their product and increase their business. Twitter is highly popular among B2B marketers. Given the number of users of the site, it did not come as a surprise discovering that Twitter is being used as a digital marketing tool by roughly 67% of all b2b businesses (Statista, 2018).

As the name suggests, Sentiment Analysis analyses textual data and mine the sentiment in it which is why the other name given to it is opinion mining. Sentiment analysis gives insight-rich information after mining meaningful information out of a large amount of data. By using the concept of sentimental analysis in business the amount of present and potential future customers can be quantified. With help of sentiment analysis, it becomes easy to have knowledge of customers' retention rate and capture the potential customer.

With such a large database of feedbacks and opinions in form of tweets, Twitter Sentiment Analysis is widely used to grow businesses. As a result, this subject has been a topic of research for a long time which is why the project attempts to find which algorithm is best suited for Twitter Sentiment Analysis [2] [3] [10].

II. Literature Review

Twitter Sentiment Analysis has been an important topic in Digital Marketing in the last decade and hence there are multiple pieces of research on the topic.

In [16], Parveen and Pandey (2016) addressed the importance of Twitter Sentiment Analysis on businesses. They used Big

data for real-time sentiment analysis of Tweets on movie reviews. The Machine Learning algorithm used was Naïve Bayes. They used Hadoop as the big data tool to analyze large amounts of tweets. They implemented the Hadoop framework for executing the Naïve Bayes algorithm. In their methodology, they used Hadoop to implement map and reduce t h e phase. And for Naïve Bayes, they used the trained SentiWordNet dictionary. In [1], the authors used the WEKA tool for preprocessing. They used StringToWordVector, which is a built-in filter to perform stepslike stemming and tokenization. They also used Feature Selection to select the potentially important feature and remove the non-relevant attributes. The paper concluded that the appropriate feature selection increases accuracy on Twitter Sentiment Analysis.

In [5], the authors tried to determine the use of linguistic features on predicting the sentiment of Tweets. They investigated the effect of Parts of Speech, Emoticons, and abbreviations on tweets. As a part of the preprocessing, they replaced the abbreviations with their actual meaning. They concluded that the Parts of Speech are not useful for sentiment analysis. Also, when the microblogging features are used the benefit of emoticon training is reduced.

In [12], the authors begin with the importance of Twitter Sentiment Analysis for digital marketers and PR agencies. They collected 300000 tweets and split them into positive, negative, and neutral tweets. They used multinomial Naïve Bayed Classifier for classification. They used n-grams and plotted their effect on the performance of the classifier. They concluded that bigrams performed better than unigrams and trigrams.

III. Proposed Methodology

NLTK dataset is divided into a training set, validation set and testing set. They are 70%, 15% and 15% of the source dataset. The frequency dictionary was build using the training set which resulted in a large dictionary. The dictionary consisted of tuples of key-value pair in the order of the words which occurred in the training dataset. Then, the frequency dictionary was sorted in the order of the frequency of the tuple so the tuple with the highest frequency was at the top and the tuple with the least frequency was at the bottom.

Machine Learning algorithms were then implemented which used the frequency dictionary for training to create a classifier.

Then the validation testing was performed. Here, training was performed on different lengths of the frequency dictionary using the algorithms implemented previously to find their accuracy by validating the prediction of the models at the validation dataset. The slice of frequency dictionary which gave the highest accuracy on validation testing is then selected. Based on validation testing, the final sorted frequency dictionary was selected which gave the highest accuracy. Then, testing was performed on the testing dataset from the NLTK. Also, testing was performed on another dataset from Kaggle. By testing on two datasets, the possibility of having a dataset bias on testing accuracy was eliminated.

IV. Twitter Sentiment Analysis

The objective of Twitter Sentiment Analysis is to classify tweets into two categories which are positive and negative. Even though it belongs to the same problem of classification of sentiment, Twitter Sentiment Analysis is very different than other sentiment analysis problems. It is because Twitter Sentiment Analysis has its hurdles over other text sentiment analyses. Because Twitter allows only a limited number of characters i.e. 280 characters in one tweet, there are added constraints to the users than any other text format. People often try to abbreviate words or skip punctuations. Also, sometimes incomplete sentences are completed in another tweet. These constraints should be comprehended and classified accordingly. Also, symbols like “#” and “@” have an altogether different meaning on Twitter and hence they should be treated accordingly. But character restrictions have an advantage: It is known beforehand that the length of characters in a tweet would not be greater than 280 characters.

A. Pre-processor:

Data is used to extract insight out of it. However, data can have noise in it. Noise is any meaningless information in data. The noise can not only increase the cost of processing but can also lead to a decrease in the efficiency of an algorithm. If noise is successfully removed, all that is left is the potentially useful data. [5] [6].

Stop-words:

Certain words add meaning to a sentence but do not contribute to the sentiment of the sentence. They are called stop-words. These stop-words could be prepositions or pronouns or any other word. They add some meaning to the sentence or are added to make the sentence grammatically correct but they can as easily be used in positive statements as they can be in negative statements. Hence, they should be removed to not waste any processing on such words [4].

Stemming algorithm:

One word can be used in different forms. E.g. hate, hatred, hateful and hatefully are all different forms of the same word

hate. The stemming Algorithm replaces all the words with their stem word by removing their suffix according to some grammatical rules [1].

For each tweet in the Twitter Dataset, perform the following steps:

1. Transform the tweet into lowercase.
2. Remove the sign '^rt' which denotes retweet.
3. Remove all types of hyperlinks.
4. Remove '#' from the beginning of the Twitter trends topic.
5. Remove all handles which begin with '@'.
6. Split remaining tweet into a list or array of words
7. For each word in the list:
    a. If the word is a stop word or punctuation delete it. If not, follow step b.
    b. Apply porter stemmer algorithm on the word and append the word in the final list of clean words.

This final list contains all the potentially meaningful and clean words. Thus, in pre-processing a tweet is converted into a vector of clean words.

B. Dataset

NLTK is currently the most famous platform for working with human language using Python program. NLTK provides a corpus of Twitter samples. This Twitter samplehas 5000 positive tweets and 5000 negative tweets. The positive tweets and negative tweets are merged to form a single dataset.

Another dataset is obtained from Kaggle named Sentiment140 which had 1.6 million tweets. The dataset is reduced to 497152 tweets with equal positive and negative tweets.

C. Division of Dataset

The NLTK dataset is divided into 3 parts. The three parts are Training, Validation and Testing. The training dataset accounts for 70% of the whole dataset whereas the validation and testing each account for 15% of the whole dataset.

D. Frequency Dictionary

Every supervised machine learning algorithm has a training dataset. Some meaning or pattern should be extracted from this dataset so it can be useful while classifying new input data. For each unique word in a positive tweet corpus, a positive frequency count should be maintained. That frequency tells the number of times a particular word has appeared in a positive tweet. Similarly, there is a negative frequency for each unique word in a negative tweet corpus which stores the negative frequency of the word. It is done by creating a dictionary with key-value pair of a tuple and an integer. A tuple is a data structure that stores more than one element. The tuple will have two elements; a word and a sentiment value where 1 is for positive and 0 is for negative. Corresponding the tuple will be an integer frequency which will tell the number of times the word appeared in the corresponding sentiment. So, a key-value pair of ("roller-coaster", 1): 30 denotes that the word "roller-coaster" appeared 30 times in the positive tweets.

E. Sorted Frequency Dictionary

It is an ordered Frequency Dictionary. In the Sorted Frequency Dictionary, the tuple-frequency pairs are sorted in order of their frequency. So, the tuple-frequency pair with the highest frequency will be at the top while the tuple-frequency pair with the least frequency will be at the bottom of the Sorted Frequency Dictionary.

F. Classification Algorithm

It is a type of supervised learning technique. It learns to map an input to an output value by learning from a large number of input-output pairs.
Classification algorithms map an input to an output where the output belongs to a set of discrete and limited categories. This classification algorithm can be used to classify a tweet as positive or negative [9].

a. Naïve Bayes:

It is a Supervised Learning Classification Algorithm. It is a probabilistic model which uses Bayes Theorem. It calculates the probability of input belonging to different classes and maps the input to the class which has the highest probability using Bayes Theorem.
Naïve Bayes classifies tweet into positive or negative based on the sum of positive probability of each word in the vectorized Tweet [16]

b. Logistic Regression:

It is a Supervised Learning Classification Algorithm. Logistic regression, also known as logistic regression analysis, is a generalized linear regression analysis model used in classification problems where the output can take one of the two possible values [18]. In the present problem, the Logistic Regression will give an output of either positive or negative [17].
It uses an activation function. The activation function used in this model is Sigmoid Function.

G. Validation Testing

The entire dataset can create a very large frequency dictionary. Such a huge dataset can cause overfitting. As a result, a portion of the dictionary is determined which gives the most efficiency. This process is called validation testing. Once the validation testing is finished, the final frequency dictionary is obtained. This final frequency dictionary is used for final testing.

H. Testing

The final model is tested against the testing data to check the efficiency of the model.

First, the testing dataset is used from NLTK. Since the model is trained and tested on the dataset with the same source i.e. from NLTK, a second dataset will be used for testing to remove bias based on the NLTK dataset. It provides a better evaluation method.

V. Negation Sentence

Negation is used for contradicting the meaning of a positive sentence by inserting a negative word like "not" into it. For e.g. "I am not happy" is the opposite of "I am happy.". Thus, inserting a negative word can change the sentiment of the sentence. At such a time removing negative words as stop-words will result in the wrong classification. Hence, negative words are not included in stop-words but the next word after a negative word is processed accordingly.

VI. Evaluation

True Positive (TP) is the number of correct predictions of positive tweets and True Negative (TN) is the number of correct predictions of Negative Tweets. Similarly, the False Positive (FP) and False Negative (FN) are wrongfully predicting tweets as positive and negative respectively.

Efficiency is the percentage of correct prediction i.e. TP and TN made by the ML model [8].

It can be calculated using the confusion matrix as:

$$\text{Accuracy} = (TP + TN)/ (TP + TN + FP + FN)$$

The confusion matrix is a matrix that gives the number of True Positive, True Negative, False Positive and False Negative.

Table 1
Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	True Negative	False Positive
Actual Positive	False Negative	True Positive

VII. Result and Conclusion

A. Sorted Frequency Dictionary

Building the frequency dictionary is one of the most important parts of the training process because this is the place that explains which phrases contribute towards positive tweets and which phrase contributes towards negative tweets.

But after building the frequency dictionary they are sorted to know the exact order of influence of words.

Taking a look at the words with highest and lowest frequency tells a lot about training purposes.

The tuple with the highest frequency is ((':(', 0.0), 2645). It signifies that the sad face emoji "☹" appeared in negative tweets 2645 times. The tuple with the second highest frequency is ((':)', 1.0), 2318). It means that the happy face emoji "😊" has appeared in positive tweets 2318 times. This shows that the happy face emoji and sad face emoji have a very influential role in determining the sentiment of the tweet [14] [15].

An interesting observation is that the last tuples in the sorted frequency dictionary have a frequency value of 1. This means that such words only appeared in the training dataset of positive or negative tweets each of size 3500 for just 1 time. Such words are the least influential and may have no or negative contribution in determining the sentiments.

This is where hyperparameter comes into the picture. The models are trained at different values of length of sorted frequency dictionary. The maximum value of hyperparameter value for both logistic regression and Naïve Bayes gave the peak accuracy came out to be 26. This shows that only the first 26% of the frequency dictionary actually positively contributes to training.

B. Validation Tables

Naïve Bayes

Table 2

Hyperparameter vs Accuracy of Naïve Bayes

Sr. No	Hyperparameter Value	Accuracy
1.	10 %	96.20
2.	20 %	96.33
3.	**30 %**	**96.67**
4.	40 %	96.00
5.	50 %	95.33
6.	60 %	95.13
7.	70 %	95.47
8.	80 %	96.20
9.	90 %	96.67
10.	100 %	97.20

Table 2 shows the changes in Accuracy of the Naïve Bayes algorithm on the validation set with a 10% increase in Hyperparameter value. A peak can be observed at hyperparameter value 30.

Table 3

Hyperparameter Breakdown vs Accuracy of Naïve Bayes

Sr. No	Hyperparameter Value	Accuracy
1.	20	96.33
2.	21	96.47
3.	22	96.40
4.	23	96.73
5.	24	96.93
6.	25	96.87
7.	**26**	**97.13**
8.	27	96.93
9.	28	96.87
10.	29	96.80
11.	30	96.67

Table 3 shows the changes in Accuracy of the Naïve Bayes algorithm on the validation set with a 1% increase in Hyperparameter value in the range 20 to 30. A peak can be observed at hyperparameter value 26. This value is considered as the final hyperparameter value for this algorithm to be used for further work.

Figure 1: Hyperparameter vs Accuracy Naïve Bayes

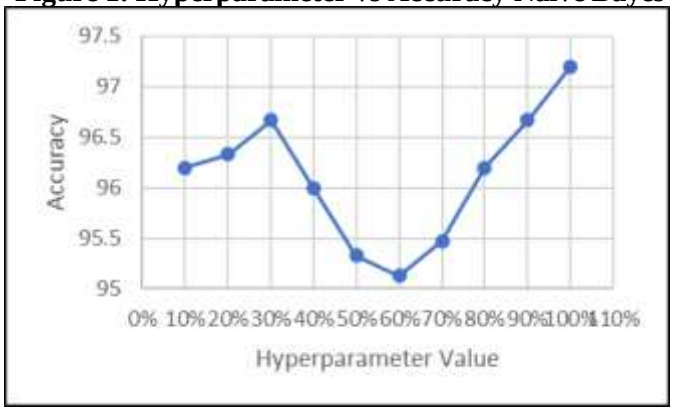
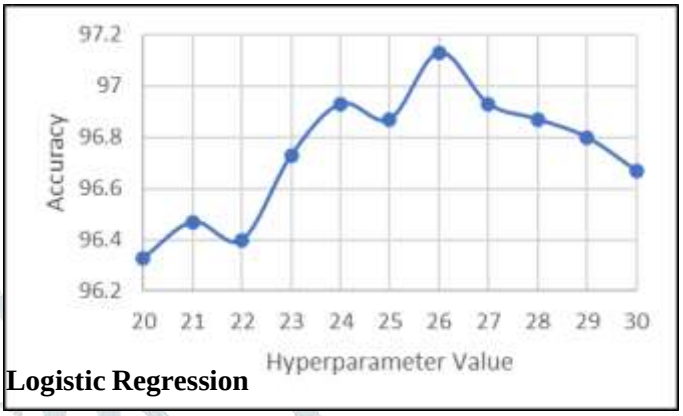


Figure 1shows the graphical representation of Table 2. peak with a minimum value of hyperparameter is visible at 30% on the horizontal axis.

Figure 2: Hyperparameter Breakdown vs Accuracy of Naive Bayes



Logistic Regression

Figure 2 shows the graphical representation of Table 3. A peak is visible at 26 on the horizontal axis.

Logistic Regression

Table 4

Hyperparameter vs Accuracy of Logistic Regression

Sr. No	Hyperparameter Value	Logistic Regression Cost	Accuracy
1.	**10 %**	**0.37278684**	**96.13**
2.	20 %	0.3726999	96.13
3.	30 %	0.37265188	96.13
4.	40 %	0.37261504	96.13
5.	50 %	0.3725746	96.13
6.	60 %	0.37258699	96.13
7.	70 %	0.37255267	96.07
8.	80 %	0.37252691	96.07
9.	90 %	0.37250057	96.07
10.	100 %	0.37247539	96.00

Table 4 shows the changes in Accuracy of the Logistic Regression algorithm on the validation set with a 10% increase in Hyperparameter value. A peak can be observed at hyperparameter value 10.

Table 5

Hyperparameter Breakdown vs Accuracy of Logistic Regression

Sr. No	Hyperparameter Value	Logistic Regression Cost	Accuracy
1.	1	0.37336284	95.53
2.	2	0.37304347	95.87
3.	3	0.37290518	95.80
4.	4	0.37295036	95.87
5.	5	0.37289663	95.93
6.	6	0.37286053	96.00
7.	7	0.37282903	96.07
8.	8	0.37280711	96.07
9.	**9**	**0.37280253**	**96.13**
10.	10	0.37278684	96.13

Table 5 shows the changes in Accuracy of the Naïve Bayes algorithm on the validation set with a 1% increase in Hyperparameter value in the range 1 to 10. A peak can be observed at hyperparameter value 9. This value is considered as the final hyperparameter value for this algorithm to be used for further work.

Figure 3: Hyperparameter vs Accuracy Logistic Regression

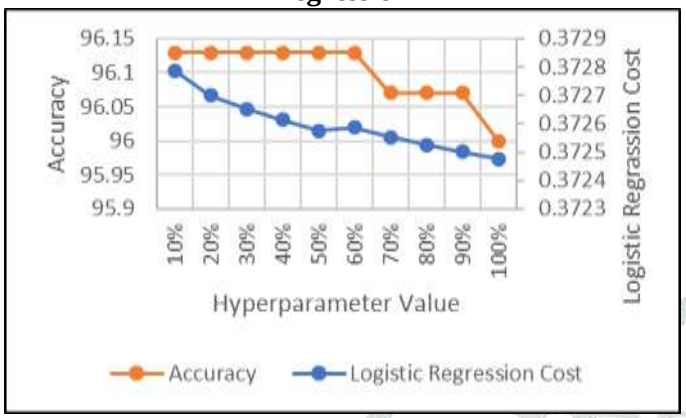


Figure 3 shows the graphical representation of Table 4. A peak with a minimum value of hyperparameter is visible at 10% of the hyperparameter value.

Figure 4: Hyperparameter Breakdown vs Accuracy Logistic Regression

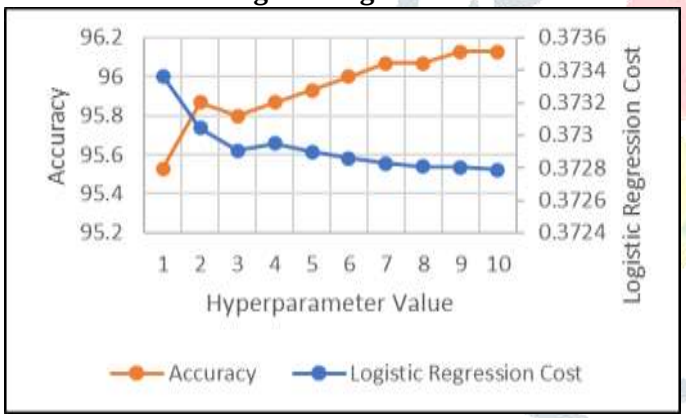


Figure 4 shows the graphical representation of Table 5. A peak is visible at 9 on the horizontal axis.

C. Final Testing

Logistic Regression

Table 6
All Testing Accuracies on Logistic Regression

Metric	Testing on Hyperparameter	Testing on NLTK	Testing on Kaggle
Accuracy	96.13	97.80	65.13

Table 6 shows the accuracy of the Logistic Regression Algorithm on three different datasets.

Figure 5: All Testing Accuracies on Logistic Regression

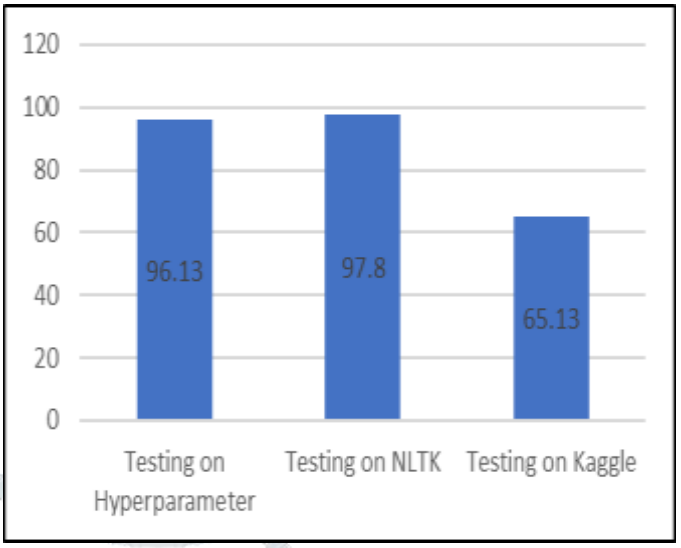


Figure 5 is a graphical representation of table 6.

Naïve Bayes

Table 7
All Testing Accuracies on Naïve Bayes

Metric	Testing on Hyperparameter	Testing on NLTK	Testing on Kaggle
Accuracy	97.13	98.66	66.74

Table 7 shows the accuracy of the Naïve Bayes Algorithm on three different datasets.

Figure 6: All Testing Accuracies on Naïve Bayes

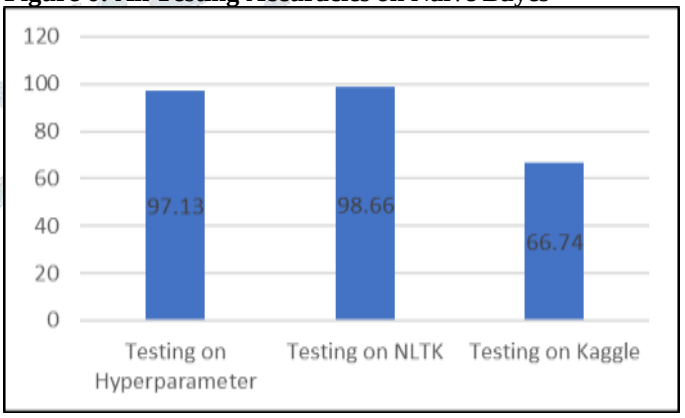


Figure 6 is a graphical representation of table 7.

VIII. Conclusion

Frequency Dictionary is the first and most important part of the training. Top elements in the Sorted Frequency Dictionary influence the results the most. The bottom elements in the Sorted Frequency Dictionary have the least or negative influence on the results.

The maximum value of the hyperparameter is 26. It shows that at maximum only a quarter of Frequency Dictionary is needed. With an increase in training dataset size, it is observed that the cost of linear regression strictly decreases. However, accuracy

doesn't strictly increase or decrease. This shows that there is a very low correlation between logistic regression cost and accuracy.

The graph of Logistic regression shows it is less sensitive to change in frequency dictionary as compared to Naïve Bayes irrespective of the preprocessor version.

Naïve Bayes performs better than Logistic Regression in both datasets. This result is consistent with the literature survey [11] [12] [13].

Both Model correctly classifies short negation tweets with high accuracy.

IX. Reference

[1] A. Krouska, C. Troussas and M. Virvou, "The effect of preprocessing techniques on Twitter sentiment analysis," 2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA), Chalkidiki, Greece, 2016, pp. 1-5, doi: 10.1109/IISA.2016.7785373

[2] Ribeiro, Filipe N., et al. "Sentibench-a benchmark comparison of state-of-the-practice sentiment analysis methods." EPJ Data Science 5.1 (2016): 1-29.

[3] D. Tang, B. Qin, F. Wei, L. Dong, T. Liu, and M. Zhou, "A joint segmentation and classification framework for sentence level sentiment classification," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 23, no. 11, pp. 1750–1761, 2015

[4] J. Zhao and X. Gui, "Comparison research on text pre-processing methods on Twitter sentiment analysis," IEEE Access, vol. 5, pp.2870–2879, 2017.

[5] Kouloumpis E, Wilson T, & Moore J. "Twitter sentiment analysis: The good the bad and the omg!" In Proc. the Fifth International AAAI Conference on Weblogs and Social Media, pp.538-541, 2011.

[6] Saif H, Fernandez M, He Y, & Alani H. "On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter". In Proc. 9th Language Resources and Evaluation Conference (LREC), Reykjavik, Iceland, pp.80-81, 2014.

[7] M. Wongkar and A. Angdresey, "Sentiment Analysis Using Naive Bayes Algorithm Of The Data Crawler: Twitter," 2019 Fourth International Conference on Informatics and Computing (ICIC), Semarang, Indonesia, 2019, pp. 1-5, doi: 10.1109/ICIC47613.2019.8985884.

[8] L. Mandloi and R. Patel, "Twitter Sentiments Analysis Using Machine Learninig Methods," 2020 International Conference for Emerging Technology (INCET), Belgaum, India, 2020, pp. 1-5, doi: 10.1109/INCET49848.2020.9154183.

[9] V. Prakruthi, D. Sindhu and D. S. Anupama Kumar, "Real Time Sentiment Analysis Of Twitter Posts," 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 2018, pp. 29-34, doi: 10.1109/CSITSS.2018.8768774

[10] R. Wagh and P. Punde, "Survey on Sentiment Analysis using Twitter Dataset," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2018, pp. 208-211, doi: 10.1109/ICECA.2018.8474783.

[11] R. I. Permatasari, M. A. Fauzi, P. P. Adikara and E. D. L. Sari, "Twitter Sentiment Analysis of Movie Reviews using Ensemble Features Based Naïve Bayes," 2018 International Conference on Sustainable Information Engineering and Technology (SIET), Malang, Indonesia, 2018, pp. 92-95, doi: 10.1109/SIET.2018.8693195.

[12] A. Pak and P. Patrick. "Twitter as a corpus for sentiment analysis and opinion mining." LREc, vol. 10,no. 2010, pp. 1320-1326. 2010.

[13] Fauzi, M. Ali, and Anny Yuniarti. "Ensemble Method for Indonesian Twitter Hate Speech Detection." Indonesian Journal of Electrical Engineering and Computer Science, vol. 11, no. 1, pp. 294-299. 2018.

[14] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for Twitter sentiment classification," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014, pp. 1555–1565.

[15] K.-L. Liu, W.-J. Li, and M. Guo, "Emoticon smoothed language models for Twitter sentiment analysis," in Proceedings of the TwentySixth AAAI Conference on Artificial Intelligence, 2012.

[16] H. Parveen and S. Pandey, "Sentiment analysis on Twitter Data-set using Naive Bayes algorithm," 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Bangalore, 2016, pp. 416-419,doi: 10.1109/ICATCCT.2016.7912034.

[17] W. P. Ramadhan, S. T. M. T. Astri Novianty and S. T. M. T. Casi Setianingsih, "Sentiment analysis using multinomial logistic regression," 2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC), Yogyakarta, Indonesia, 2017, pp. 46-49, doi: 10.1109/ICCEREC.2017.8226700.

[18] Y. Wang, Y. Ou, X. Deng, L. Zhao and C. Zhang, "The Ship Collision Accidents Based on Logistic Regression and Big Data," 2019 Chinese Control And Decision Conference (CCDC), Nanchang, China, 2019, pp. 4438-4440, doi: 10.1109/CCDC.2019.8832686