Hate Speech Detection Using Machine Learning and Natural Language Processing

N. Sai Aryan
Internal Guide
Department of Computer Science and Engineering (AI & ML)
CMR Engineering College

Ms. B. Revathi
UG Scholar
Department of Computer Science and Engineering (AI & ML)
CMR Engineering College
bigullarevathi@cmrec.ac.in

N. Sumanth Reddy
UG Scholar
Department of Computer Science and Engineering (AI & ML)
CMR Engineering College

M. Shanmukh
UG Scholar
Department of Computer Science and Engineering (AI & ML)
CMR Engineering College

P. Lokesh
Department of Mechanical Engineering, CMR Engineering College,
Hyderabad, Telangana 501 401, India

Abstract

The escalating incidents of hate and offensive words or speech being used on public social media platforms may cause damage to social media environment safety and prevent the welcoming of new users. Lynchings of Muslims in India, abuse of marginalized groups like Black, female, and LGBTQ users, and the storming of the US Capitol have all been associated with hate speech proliferation on Twitter. In this current study, we offer a comprehensive model with efficient evaluation metrics and better performance in comparison to remaining hate speech detection models.

The model is a classifier with hate, offense, and neither class. It uses Natural Language Processing techniques - TFIDF, polarity scores, and doc2vec embedding on the text by training them using Logistic Regression, Naïve Bayes, Random Forest Classifier, linear SVC and analysing them using the classification evaluation metrics accuracy scores, confusion matrices, phi coefficient. The model training majorly focuses on feature engineering of the text data using NLP techniques and the hyperparameters tuned for the models and estimators. The results came out to be in favour of logistic regression and linear SVC estimators with an efficient feature matrix of polarity scores, doc2vec embeddings, and count features of the dataset with 81% and 86% accuracy scores.

Keywords: *Hate Speech Detection, Natural Language Processing, Logistic Regression, Linear SVC, Confusion matrix, phi coefficient.*

I. INTRODUCTION

Hate speech detection leverages machine learning to analyze social media content and identify linguistic markers of hate or offense. By applying natural language processing, these algorithms can categorize text as hate speech, offensive language, or neither. This empowers

developers to moderate content that violates standards of respectful discourse. Effective hate speech detection allows social platforms to proactively remove abusive posts and accounts. Automated content analysis promotes inclusive online communities by eliminating harmful speech that may alienate users.

In today's era, users expect safe, healthy environments across online services. Social media enables opinion sharing without hesitation. However, some content offends certain users while seeming inoffensive to others. Offended users may migrate from the platform. Additionally, children's social media use risks impacting their development. Sophisticated content analysis is required to maintain inclusive online communities. Emerging hate speech detection can identify linguistic markers of offense. Machine learning models can automatically flag abusive posts for removal. This promotes constructive digital discourse by eliminating alienating speech.[1]

The lack of clear delineation between hate speech and offensive language poses challenges. Words offensive to some may not universally offend. Hate speech harms all of society while offensive language targets specific groups. Indiscriminate flagging of offensive posts as hate risks over-enforcement. Slang taken out of context may be misclassified. Precise hate speech detection is vital as improper labelling can erroneously erase benign social media content. Sophisticated natural language processing is required to analyse linguistic nuances. Machine learning models should distinguish between language that alienates user groups and broader hateful ideologies that damage community integrity. Accurate classification enables moderation that balances open expression with the removal of truly abusive posts. [2]

The model is a classification model that is being trained by using multiple machine-learning classifiers namely logistic regression, random forest classifier, gaussian naïve bayes, and linear svc. These estimators are being used based on the potential features and the size of the dataset. The dataset has a text format of tweets, a count of hate words, and offensive words, and also the words that are complex but do not belong to any of the classes are classified as neither.[3]

Natural processing is the way in machine learning which enables better communication between a model and text data as the text data cannot be directly fed to the model. The polarity scores of the text, Term frequency, and inverse document frequency (TF-IDF) for a word are used to represent the relevance of words in multiple tweets, the doc2vec embedding model helps in vector representation of the text data providing a neural network model with trained and adjusted nodes, and also the counts of the hate and offensive word infers a major toll on the efficiency of the model.[4]

The previous works on hate speech detection with multiple classes as hate, and offensive, and neither has produced a proper model but have limited performance metrics and access to the hate data as the dataset has a very low portion of tweets labeled as hate. Many of the previous works have been focused more on classifying data as hate or non-hate not considering the relevance of some words to the offensive but not to hate.[5]

In this paper, our results have proved to be positively working against the imbalance in the dataset for hate class and the added feature set of the count of hate and offensive words mathematical equation has proved to be increasing the efficiency of the model with fine-tuned parameters of the models.[6]

II. RELATED WORK

Recently there has been a surge in the research in the field of hate speech detection to implement it in the most efficient way in social media applications and websites so that there is no discomfort for its users. Most of the models are limited only to classify hate or non-hate but the factor that even the offensive terms that are only hate towards certain set of people are also being identified as hate speech. For example, the word n*g*a is a hate word for African people, but it's a casual word for African Americans likewise the word p*s*y can be offensive towards the female gender but is not in any way affected towards male gender, this way the words have to be classified into three different classes.

In the past few years, there has been remarkable work done for hate speech detection by implementing the neural networks to be used to train the model, in which the model adjusts its weights for every input point and proving that it is possible to use to neural networks, random forest classifier, and k – NN classifier which have proved to the most basic and efficient algorithms (Bishop Raj Majumder, Bibek Kumar Ghosh, Farazul Hoda, P. Preethy Jemima. 2022), whereas the dataset of hate speech detection does not have a class differentiating the hate speech from offensive speech, which has proved not so efficient on vivid datasets. The used methodologies were Random Forest, NLP, sentiment analysis, ANNs, RNN, LSTM, and Back Propagation Neural Networks.[1]

Previous works also include the model with the same dataset being used with giving importance to multiple NLP utilizing tools such as TFIDF vectors, Penn Parts of Speech (POS), sentiment lexicon (Hutto, C. J., and Gilbert, E. 2014), and Flesch Kincaid reading scores for single sentences were used as major features. The models used for training are the most prominent classifiers in the world of classification logistic regression, random forest classifier, gaussian naïve bayes classifier, and linear SVM that works with different kernels (Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber. 2017).[2]

The problem raised among the model was that the dataset had an imbalance towards the hate sample data, which would result in underfitting of the model in the realm of hate class detection and the testing cannot be done efficiently to know the performance of the model on the true unseen data. Even though the L1 and L2 regularizations were used to address this issue, it won't be sufficient enough to produce a model with a balancing in the under-available class samples.[3]

The usage of sentiment polarity scores and subjectivity semantic orientation of the text to generate a lexicon-based approach (Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien and Jun Long. 2015) also has been a useful methodology to detect hate speech in the previous decade but instead of a feature set for the model to learn the lexicon may not be efficient enough as the language of hate can be expressed on multiple ways and the tokens of words in the lexicon and their sentiment scores cannot be utilized to the extent of the precision.[4]

As more and more hate and offensive content is posted online over multiple social media opinion platforms such as Twitter, it has become a matter of issue to detect and handle these kinds of tweets or comments to maintain a healthy user social environment there has been a lot of research going on in the past few decades and mostly they have been more focusing upon hate and non-hate but the fact that not all the hate classified tweets are hate speech for every user, instead it's enough to hide these content form the user who might be

targeted due to those words such as n*g**, j*w**h, w**te, tr*n*, etc.[5]

To make this possible we have proposed an approach utilizing multiple feature sets integrated with NLP techniques and various machine learning classifying estimators to produce a model with greater classifying efficiency without any effect of the low count of the hate class samples in the dataset. The most prominent additions to the model feature set include TFIDF hyperparameters experimentation tuning, sentiment analysis polarity scores, and a doc2vec embedding model from genism to represent the text data of the tweets in the form of vectors generated through a pre-trained doc2vec embedding model with a distributed bag of words(PV-DBOW) and using the numerical features given in the dataset such as counts of hate words, offensive words in a given tweet can be utilized to generate mathematical values to make the different classes samples unique.[6]

The data-preprocessing of the dataset tweets includes cleaning the data by removing unnecessary information from the tweets, which are not so useful in the training of the model and also may result in wrong learning of the model. The fine-tuned parameters of the different algorithms used in the feature engineering for the model have generated various kinds of effects on the performance metrics of the model and also have played a crucial role in improving the accuracy and confusion matrix of the model.[7]

The effective hyperparameter tuning of the models as well as the feature models has made the vivid representations of the text in different formats possible so that the feature set can be fed to the model with different estimators and analyze which set has the most efficiency and performance metrics for the classification task.[8]

III. METHODOLOGY

Hate speech detection doesn't only comprise the task of classifying the text into hate and non-hate but also it keeps an effort to detect the words that are hate text and the ones that are offensive in a sense too. The task of classifying both types of text signifies the change on the issue that offensive speech is not hate speech to every user but it only relates to the set of people that may be offended by it.

The dataset that is being used for this model is collected from Kaggle and is crowd-sourced data, due to which it comprises more casual and liberal texts that are real to their core, and also it generalizes the text data so that it can be made to perfectly work with the models and algorithms. The dataset contains 6 different important features namely count(sum of hate words, offensive words, and the words that are negative but are neither hate nor offensive), hate_speech(number of hate words in a given text), offensive_language(number of offensive words in given piece of text), neither(words which are neither hate nor offensive), class(the label that is given to the text), tweet(the most useful feature of the dataset that is the text data from the social media services).

There are 3 different classes available in the dataset for the tweets, namely 0 for hate speech, 1 for offensive speech, and 2 for neither. The dataset analysis explains that the samples of hate speech with class 0 are less in number as the hate speech only comprises 5.6% of the whole dataset. The model is efficient enough in adjusting to the dataset imbalance by manipulating the weights in the estimator algorithms for the imbalance does not affect the classification process. The dataset preprocessing includes the removal of stop words from English contiguous spaces, leading and trailing white spaces, links, user

mentions, punctuations, and numbers, and tokenizing the tweets to simplify these words lemmatization is used which tokenizes the text by maintaining the true dictionary word.

After the data is preprocessed and the tokens of words are acquired, the next process includes the training of the models by the NLP and embedding algorithms in order so that the model gets unique features of the text so that they can learn through the representations of the text. As the process is a supervised learning technique, the model tried to match the feature set data to synchronize with the class labels and train the model according to the tweet labels. The features used for the classification process are TFIDF vectorization, sentiment analysis polarity scores, doc2vec embedding representation, and numerical mathematical features of the count of different types of words in the tweet.

All of the feature models have been fine-tuned to acquire the precise hyperparameters at which the model works with a high-performance rate. The TFIDF vectors are used so that the relevance of a word with high frequency in a single document can be compared with the count of the same word in other documents or tweets calculated its TFIDF score and added to the TFIDF vector if it falls in the range of threshold values of the vector which are min_df and max_df. The max_features is used to add as many words that can be useful to the model to learn. The optimal values for the TFIDF are the n_gram range of (1,2), max_df of 0.75 and min_df of 5. The trained model of TFIDF with these parameters can be of utmost useful to increase the accuracy and precision of the model.

The additional feature added to the TFIDF is the sentiment analysis polarity scores that are used to define the mood or the state of the piece of text according to the score and as a piece of extra information, it returns the percentage of all the 3 polarity score values which are positivity, negativity, and neutrality of the text. The sentiment analysis cannot be performed on the tokens of words as it may result in the wrong presentation of the polarity scores and the original tweets contain the URLs, mentions, etc. as unnecessary information. These tweets will be processed through regular expression patterns so that they can be replaced by a placeholder and the count of these are noted so that they can be added to the polarity scores feature matrix and the model learns about these placeholders to be left out of the polarity score values as they are provided along with their count values.

The third feature added to the feature set is the doc2vec word embedding model vector representations which is an NLP technique used to represent the text in the form of vectors by training the doc2vec model with a distributed bag of words algorithm (PV-DBOW) and a neural network is used to adjust its weights according to the PV DBOW algorithm and gets trained by this model. Doc2vec is an extended feature to the word2vec with an enhanced presentation of the text as a document to the model. It is an unsupervised learning technique and also utilizes this to train the model. When unseen data is passed through this model it generates an infer vector representing the vector representation of the document. These vectors can be used by the model to understand the semantic features of the text and the word embeddings in the text so that the context of the document can be understood by the learning model.

The fourth and last feature is to utilize the already provided numerical features from the dataset, which are counts of hate words, offensive words, and words that are complex but do not belong to any other class. These numerical values are processed through two of the mathematical equations applied to the count values namely, statistical measures of the words and the indicator variables for the 3 different classes. The mean is applied over the hate

words count as the count of hate class samples is low in number and the standard deviation of the offensive class is used to depict to the model that the deviations in the very large part of the given dataset are to be considered during the learning and the median is applied on the neither as their count resides between the counts of hate and neither word. Another numerical equation would be an indicator variable which indicates that if the given record of text has a hate count >0 then it would be indicated that hate of this tweet is true and the same for the other counts of words.

IV. RESULT AND DISCUSSION

Each classification algorithm along with different combinations of feature sets is scrutinized across multiple performance metrics, shedding light on its efficacy in classification of the new unseen text into hate, offensive, or neither class. The Accuracy metric serves as a holistic gauge, portraying the general correctness of each algorithm's predictions across the entire dataset. A higher accuracy score implies a more proficient algorithm in making accurate classification. Phi coefficient and Recall measure the algorithm's adeptness in correctly identifying instances of hate among the actual hate occurrences. This metric provides insights into the algorithm's ability to minimize false negatives, ensuring robust performance in capturing true positive cases. Specificity delves into the algorithm's capacity to accurately identify instances that do not correspond to hate and offensive, gauging its precision in recognizing non-hate speech. This metric is crucial for assessing the algorithm's ability to avoid false positives. The confusion matrix for every feature set accurately depicts the true positive and false positive percentages predicted for all the classes.

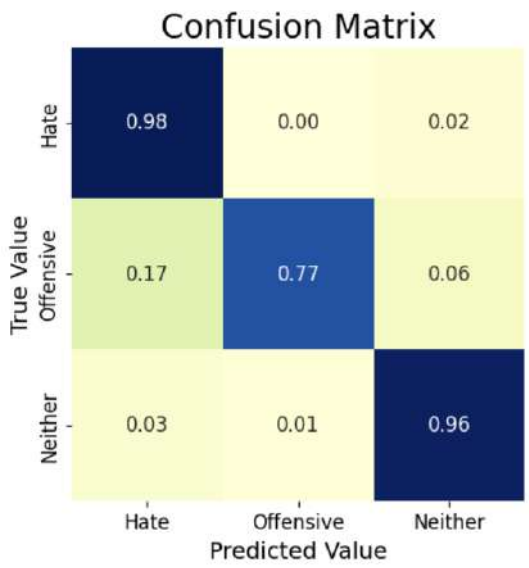
The outcome of the project and the model that is compatible with the imbalance in the hate speech class samples and produces a maximum true positive rate for all the three classes is achieved by feature engineering of three different features in a combination which are features matrices of F2, F3 and F4 namely polarity scores, doc2vec representations and numerical features of count data. These have proved to be effective features set for the model to be trained with a higher accuracy score of 0.87 and a phi co-efficient of 0.71 value. The performance metrics of the model are in Table 1.

Table 1: All feature metrics

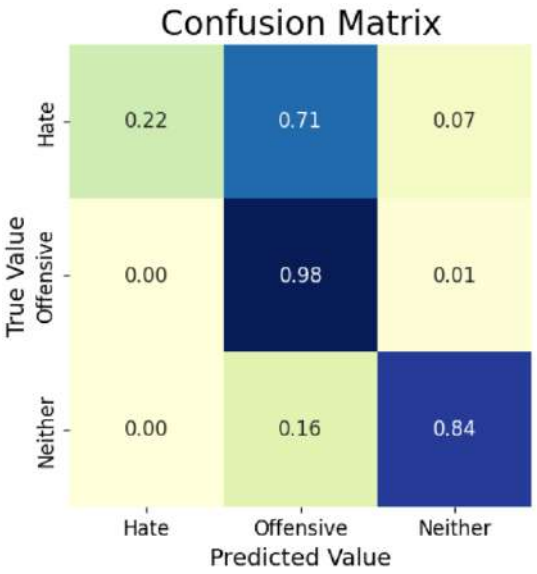
Algorithm	**Logistic regression**	**Random Forest**	**Naïve Bayes**	**Linear SVC**
Accuracy	0.81	0.92	0.89	0.87
Precision	0.92	0.92	0.90	0.91
Recall	0.81	0.92	0.89	0.87
F1-score	0.84	0.90	0.89	0.88
MCC	0.67	0.76	0.72	0.71

The major performance metrics of a multiclass classification model are not only the accuracy score, f2 score, and recall score, but also the confusion matrix which depicts the true positives and false positives of the different classes in the range of 0 to 1 and the color scaling with yellow green blue, blue representing the true predictions and yellow representing the false predictions and green is the standard medium range of predictions. The feature set has

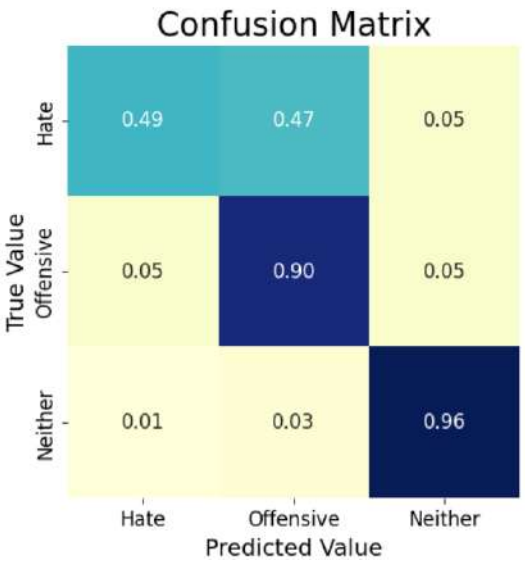
greater accuracy and other metrics for the random forest and naïve bayes classifiers, but the confusion matrix for all the estimators with a feature set of polarity scores, doc2vec vectors, and numerical features are as follows:



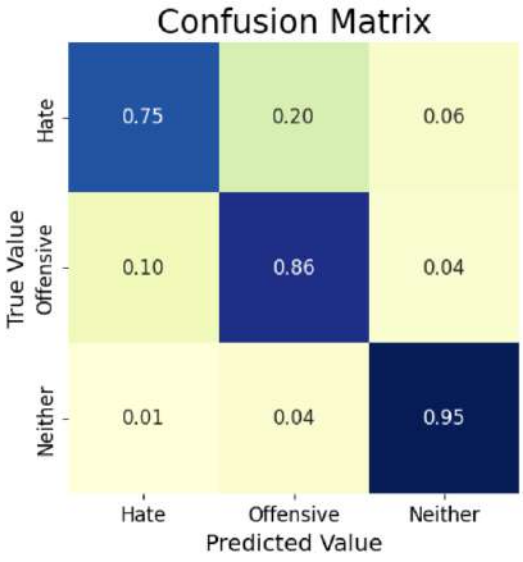
Confusion Matrix - Logistic Regression



Confusion Matrix – Random Forest



Confusion Matrix - Naïve Bayes



Confusion Matrix – LinearSVC

The accuracy scores for the random forest and naive bayes classifier are greater compared to the rest estimators, but the confusion matrix results portray the reason for their higher accuracy scores. The true positives for offensive and neither class is very high as compared to the hate class, hence their accuracy scores are high. The important task of this paper is to enhance a model that can adjust with the imbalance of the hate class samples and improve their true positive rates, which can be clearly seen in the Logistic regression and LinearSVC estimators. The confusion matrix of the logistic regression is good but has an overfitting issue and has low accuracy score as compared to LinearSVC, hence the LinearSVC estimator with a true positive for hate being 75%, offensive being 86% and

neither class being 95% is the best accurate model with higher and balanced classification metrics for each class has to be the model to be put in use for hate speech detection.

V. CONCLUSION AND FUTURE SCOPE

The hate speech detection model is the most important aspect of social media as it may supports in maintaining the healthy social environment for the users. As this task cannot be always done by human-power the machine learning is used to develop a model to perform this task efficiently. The disadvantage of having a low count of hate speech to train the data is been resolved by using multiple NLP techniques used as feature set. The performance metric have been better as compared with the previous models and also the usage of extensive numerical features from count of hate and offensive words has resulted in a model with an accuracy of 87%. It is crucial to learn that the most efficient feature set and estimator combination is polarity scores, doc2vec representations, and numerical features trained by using the LinearSVC estimator. Hence the hate class unseen data can also be classified efficiently up to 75% of the test data.

The gradual increase in the effective classification task by the model through the feature engineering explains the fine tuning of the hyperparameters. After testing the dataset over multiple combinations of feature sets the concluding point comes out to be that the TFIDF vectorized scores being the word eliminating process does not contribute much in the classification task and is responsible for the decrease in classification of hate speech in most of the models and hence it is not a part of the final model. The sentiment scores and doc2vec resulted in appropriate text representations in the numerical format for the estimators to learn and perform with maximum performance metrics. Accuracy score, phi coefficient and the confusion matrix have helped a lot in comparison of different models and select among them.

Future enhancements to the hate speech detection model would be implementing the advanced NLP techniques like pre-trained models so that they can be trained over the dataset to produce a start-of-art performance model and at the same time the word embeddings and the context capturing will be effective as compared to the rest. The use of deep neural networks such as LSTM can be more effective to increase performance. The neural networks work performing the weight adjustments along all the records of the dataset making it simpler for the model to learn through the imbalance in dataset. The count of hate speech and offensive language can be used efficiently through statistical and mathematical featuring, to allow the model learn through the frequency of the words matched with the counts of hate and offensive words.

References

[1] Bishop Raj Majumder, Bibek Kumar Ghosh, Farazul Hoda, P. Preethy Jemima. Conference on “Hate Speech Detection using Machine Learning”. IEEE Xplore 2022.
[2] Thomas Davidson, Dana Warmsley, Michael Macy, IngmarWeber. Conference on “Automated Hate Speech Detection and the Problem of Offensive Language”. 2017.
[3] Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien, and Jun Long. 2015. A lexicon-based approach for hate speech detection. International Journal of Multimedia and Ubiquitous Engineering, 10(4):215–230.
[4] Bird, S.; Loper, E.; and Klein, E. 2009. Natural Language Processing with Python. O’Reilly Media Inc.
[5] Djuric, N.; Zhou, J.; Morris, R.; Grbovic, M.; Radosavljevic, V.; and Bhamidipati, N. 2015. Hate speech detection with comment embeddings. In WWW, 29–30.
[6] Hutto, C. J., and Gilbert, E. 2014. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In ICWSM.

[7] Kwok, I., and Wang, Y. 2013. Locate the hate: Detecting tweets against blacks. In AAAI.
[8] Nobata, C.; Tetreault, J.; Thomas, A.; Mehdad, Y.; and Chang, Y. 2016. Abusive language detection in online user content. In WWW, 145–153.
[9] Wang, W.; Chen, L.; Thirunarayan, K.; and Sheth, A. P. 2014. Cursing in english on twitter. In CSCW, 415–425.
[10] Warner, W., and Hirschberg, J. 2012. Detecting hate speech on the world wide web. In LSM, 19–26.