Hate Speech Detection in Social Media

Praveenraj Velusawami
pvelus2@uic.edu
University of Illinois at Chicago
Chicago, Illinois, USA

Shobana A. Iyyasami
sarumu6@uic.edu
University of Illinois at Chicago
Chicago, Illinois, USA

ABSTRACT

Hate speech has lately emerged as one of the most well-known problems due to the enormous rise in internet and social media usage among individuals. Hate speech in the press and on social media has increased manifold during elections, social unrest and even during the most current COVID outbreak. if left unchecked, this has the potential to stoke large scale violence. As a result, it is now crucial for the social media giants to identify and de-platform these terminate these dangerous contents. However, there are several situations where the line of demarcation between "hate speech" and "offensive speech" is not conspicuous. Under such circumstances, Artificial Intelligence can be used to create tools to aid in this process. With the help of machine learning models, we have developed an efficient system to classify hate speech into three categories i.e. hate speech, offensive and neither hateful nor offensive. We have understood and implemented SMOTE, a technique that eliminates the inherent bias in the dataset in an attempt to accurately identify "hate speech". We test this system on a real-world Twitter dataset.

KEYWORDS

hate speech, offensive speech, artificial intelligence, machine learning, SMOTE, bias

ACM Reference Format:
Praveenraj Velusawami and Shobana A. Iyyasami. . Hate Speech Detection in Social Media. In *Proceedings of* . ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

With the advent of technology and increased consumption of content on social media, there has been a steady increase in the instances of online harassment and cyberbullying. More often than not, these crimes stem from a common root - hatred. Hate crimes should not be treated as isolated incidents. A deeper dive into these crimes reveal that the single point of their origin is hate speech. It is common knowledge that social media holds access to a wide range of audience belonging to different countries, age groups, ethnicity, religion, race and gender etc. With very little or no consequences to boot, it is very easy to propagate hatred on social media anonymously. While it is important that free speech be ensured, it is equally necessary to have checks and balances to make sure that free speech does not transcend into hate speech. Hate speech has very dire consequences which include but are not limited to targeted harassment of the subject of hatred often leading to violence and sometimes, even resulting in death. Inflammatory content on social media become viral within a short span of time. This is a warning bell in itself and shows that there is a large consumer base for such content. While this might seem insignificant, such contents as seen in the past, have provoked events that and incited communal and civil strife. With a delicate social fabric prevailing over the world, it is imperative to monitor and censor hate speech on social media platforms. However, there is a catch. All hate speech is offensive in nature. But the converse need not be true. What differentiates ‘hate speech’ from ‘offensive speech’ is its targeted nature of making a particular group based on their class, race, religion, gender and/or language an object of ire for other homogeneous groups who identify differently and subscribe to different ideologies and beliefs. ‘Offensive speech’ on the other hand, does not have any ulterior motive other than simply offending the user. While ‘offensive speech’ is repulsive, it does not perpetrate harm at a large-scale like ‘hate speech’ does. Therefore, it is of utmost importance to have ‘hate speech’ distinguished from ‘offensive content’. This research seeks to do exactly the same by treading on the thin boundary separating the two and designing an intelligent system that does away with the inherently prevailing bias towards either of these speeches and accurately identifies ‘hateful content’.

2 RELATED WORK

As Cyberbullying kept getting mainstreamed, AI researchers worked to develop systems that aid in social media policy and decision making like Pete Burnap et al who employed classification and statistical regression models on data collected and curated by crowdsourcing hate speech annotators. This was further extended in a specific direction by Thomas Davidson et al who made use of Liblinear SVM and incorporated that for temporal classification and native language identification for hate speech detection in English language. This paper saw the use of surface n-grams embedded into skip-grams for identifying surface features.

3 PROBLEM STATEMENT

In accordance with the aim of this project, the problem statement can be actualised into a standard Classification Model. Since the goal is to build a socially responsible AI system that can distinguish hate speech from non-hate speech, it is imperative that the system will, at a very basic level, identify what qualifies as hate speech. A Classification model is best suited for this task and that will be the line of work we will perform in this project.

4 THE DATASET

The source of our data is Hatebase which is a repository of multilingual hate speech lexicons. Hatebase was built to assist companies, government agencies, NGOs and research organizations moderate online conversations and potentially use hate speech as a predictor for regional violence.

4.1 Data Collection

A set of lexicons were derived from www.hatebase.org and applied on a Twitter dataset consisting of 24783 tweets. This was then distributed among a number of CrowdFlower users who were tasked with labeling the tweets as either 'hate speech' or 'offensive language' or 'neither'. Curated by Davidson et al, this dataset was then made available in public by them for their work on hate speech detection.

4.2 Data Description

ATTRIBUTES: Index, Count, Hate Speech, Offensive Language, Neither, Class and Tweet

(1) INDEX: Index of each record.
(2) COUNT : Number of CrowdFlower users who coded each tweet.
(3) HATE SPEECH : Number of CrowdFlower users who judged the tweets to be hateful.
(4) OFFENSIVE LANGUAGE : Number of CrowdFlower users who judged the tweets to be offensive.
(5) NEITHER : Number of CrowdFlower users who judged the tweets to be neither offensive nor hateful.
(6) CLASS : Class label as classified by majority of CrowdFlower users where 0 indicates hate speech, 1 means offensive language and 2 stands for neither hateful nor offensive.
(7) TWEET : Tweet which is in the form of text.

4.3 Data Pre-processing

The tweets were cleaned and pre-processed. The pre-processed data was then used to build the baseline model. The following steps were performed as a part of pre-processing:

(1) Expansion of clitics
(2) Sentence tokenization
(3) Removal of punctuation
(4) Removal of stop words
(5) Conversion of sentences to lowercase
(6) Removal of unwanted tokens like T and RT which indicate tweet and retweet

$OriginalTweet$: "!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. amp; as a man you should always take the trash out..."

$Pre-processedTweet$: mayasolovely as a woman you shouldn't complain about cleaning up your house amp as a man you should always take the trash out

5 EXPLORATORY DATA ANALYSIS

Our dataset has 24783 records in total and 7 columns including the processed tweets obtained in the data cleaning process (section 4.3). Here are a few inferences we have after conducting an EDA on our data.

- On an average there are close to 3 annotators for each tweet i.e. each record.
- Minimum and maximum number of annotators per tweet is 3 and 9 respectively.
- Tweets identified as offensive constitute a major chunk of the data we have. The number of tweets identified as offensive, hate and neither are 19190, 4163 and 1430 respectively.
- Around 77 percent of the records are of offensive tweets.

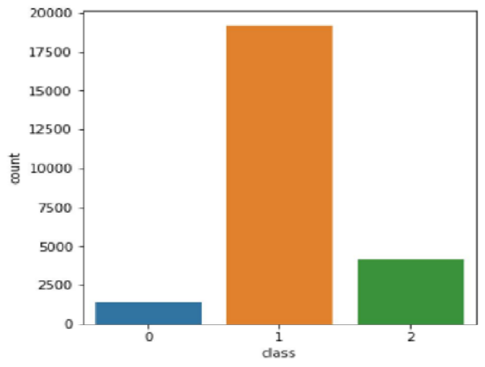


Figure 1: EDA: Tweet distribution by class where 0 indicates hate speech, 1 stands for offensive lan

6 MODELS

6.1 Baseline Model:Multinomial Logistic Regression

The pre-processed tweets were used to develop the baseline model. We made use of the Term Frequency-Inverse Document Frequency (TF-IDF) vector. A word combination of bigrams was chosen. A tf-idf matrix generated from the pre-processed tweets was used as the input to our baseline model. The dataset was then split into training and testing sets with 75 percent of the data being used for training and 25 percent of the data being used for testing. A Multinomial Logistic Regression model was then built for the same. It is another algorithm that can be used for classification. This algorithm puts one class against all other classes and runs a binary classifier. It is called One Versus Rest algorithm precisely for this reason. The probability of each class is returned during execution and the classifier chooses the class with the highest probability.

The baseline model has an overall accuracy of 90.07 percent, precision of 0.88, recall of 0.90 and f1-score of 0.89 respectively. The Precision of individual classes was noted to be the highest for class "Offensive Tweets" at 0.92 and lowest for "Hate Tweets" at 0.44. Similar trend was observed for Recall and Accuracy with "Offensive Tweets" having a Recall and F1-Score of 0.96 and 0.94 respectively while "Hate Tweets" had a Recall of 0.21 and F1-Score of 0.28.

6.2 Support Vector Machine

A Support Vector Machine uses a separating hyperplane to discriminate between data points and classify them. It classifies data points into two classes at a time. Given a set of training data, it results in an optimal hyperplane which categorizes data into the classes to which they belong.

6.3 Decision Tree Classifier

Decision trees are widely used in data mining of all sorts. With a top down approach, a tree structure is formed by dividing the features into uniform subgroups using divide and conquer, having the leaves as the final outcome.

6.4 Stochastic Gradient Descent Classifier

Gradient Descent is an optimization technique where parameters of the model are tweaked in order to minimize the cost function. When the cost function is calculated using a single random sample of the data then the algorithm is called stochastic gradient descent. While dealing with large datasets, the SGD Classifier performs in a faster and morev efficient manner.

6.5 Multi-Layer Perceptron Classifier

A perceptron classifier is a linear algorithm that can be applied to binary classification problems. It is the simplest type of neural network. It consists of single node or neuron that takes a row of data as input and predicts a class label. The weights of this neuron are fairly easy to interpret. Suppose that a feature gets a zero weight. Then the activation is the same regardless of the value of this feature. So features with zero weight are ignored. Features with positive weights are indicative of positive examples because they cause the activation to increase. Features with negative weights are indicative of negative examples because they cause the activation to decrease. Weighted sum of inputs is used to compute the Activation. Activation = (weights x inputs) + bias where bias is set to 1. As this algorithm is inherently error-driven, there is no need to update the parameters.

6.6 XGBoost Classifier

XGBoost Classifier is an classification algorithm that is specially designed for the purpose of speed and improved performance. The concept of boosting is an ensemble technique where new models are added to existing models to correct the errors made by them. Sequential addition of models is performed until no further improvements can be made.

6.7 KNN Classifier

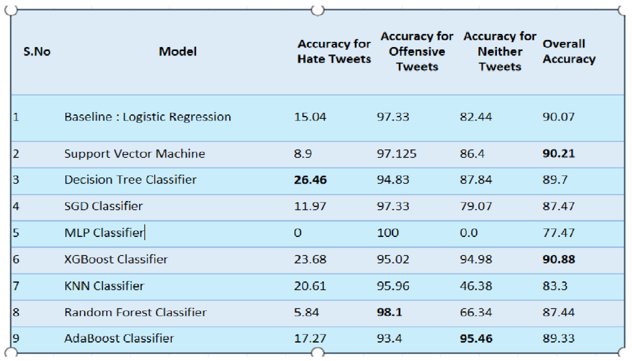
K Nearest Neighbour or KNN as it is called, is a non-parametric supervised machine learning model that focuses on instance-based learning. The hyperparameter K indicates the number of neighbours. The number of neighbours is initially set to 3.

6.8 Random Forest

Random Forest builds an ensemble of decision trees and trains them with "bagging" method. This method votes for the best decision out of the ones decided by all the decision trees.

6.9 AdaBoost Classifier

AdaBoost sequentially grows decision trees and punishes those samples that were incorrectly predicted by assigning a larger weight to them after each round of prediction. This incentivizes learning from previous mistakes of the model. Finally, the prediction that is weighted majority vote is given as the output.



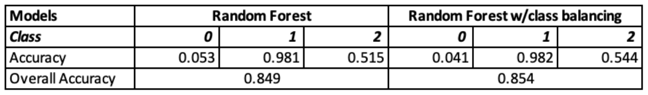
S.No	Model	Accuracy for Hate Tweets	Accuracy for Offensive Tweets	Accuracy for Neither Tweets	Overall Accuracy
1	Baseline : Logistic Regression	15.04	97.33	82.44	90.07
2	Support Vector Machine	8.9	97.125	86.4	**90.21**
3	Decision Tree Classifier	**26.46**	94.83	87.84	89.7
4	SGD Classifier	11.97	97.33	79.07	87.47
5	MLP Classifier	0	100	0.0	77.47
6	XGBoost Classifier	23.68	95.02	94.98	**90.88**
7	KNN Classifier	20.61	95.96	46.38	83.3
8	Random Forest Classifier	5.84	**98.1**	66.34	87.44
9	AdaBoost Classifier	17.27	93.4	**95.46**	89.33

Figure 2: A summary of performance of all models used

6.10 Building A State-of-the-Art Model by Exterminating Bias

Through our results we have come to the conclusion that the dataset being heavily imbalanced has contributed to the low accuracy on the minority classes. One of our initial ideas was to look for a dataset which is balanced which would help us build a model that has a high accuracy. However, we believe that the core of this dataset is to segregate offensive tweets from hate tweets which not many datasets have done previously. We also see this as an opportunity to explore imbalanced classification and find ways to improve the accuracy for the minority class. It is important for us to find ways to mitigate this to have a robust model that can be relied upon. Random Forest classifier can help us get an improved accuracy by letting us set the class weights to 'balanced' in the RandomForestClassifier module of the 'sklearn' toolkit [1]. This basically sets equal weights for each class thus not leading to a bias towards a specific class. Although we did not achieve a good accuracy with the Random Forest classifier, we tried this again balancing the class weights.

6.10.1 Random Forest classifier with balanced class weights: The result can be found in Figure 3. We can see a slight increase in overall accuracy and class 2 accuracy. However, the accuracy for class 0 has actually decreased while class 1's accuracy has almost remained the same. We have achieved better overall and minority class accuracies using other models earlier. Hence, we decided to move on from the Random Forest classifier.



Models	**Random Forest**			**Random Forest w/class balancing**		
Class	***0***	***1***	***2***	***0***	***1***	***2***
Accuracy	0.053	0.981	0.515	0.041	0.982	0.544
Overall Accuracy	0.849			0.854		

Figure 3: While the overall accuracy has increased slightly, we cannot see any fruitful results for the minority classes. Class 0,1 and 2 represents Offensive, Hate speech and Neither respectively.

6.10.2 Balancing the data. Another way to handle the disparity in accuracy is to handle the data itself. We can get rid of the imbalance in the data either by undersampling the majority class samples or oversampling the minority class samples. Since we have close to 25,000 samples in our dataset and the difference between the majority and minority class samples being so large, we believe that undersampling the majority class would lead to significant loss of data which is not ideal for model training. Hence, we decide to oversample the minority class. This can be achieved in two ways. We can either duplicate the samples in the minority class to match the majority class or synthesize new samples for the minority class. The former would lead to loss of variation in the data which is crucial for our problem statement. Hence, we look for ways to synthesize new samples for our minority classes. Among a few popular techniques, we choose SMOTE (Synthetic Minority Oversampling Technique) [2].

6.10.3 SMOTE. SMOTE or Synthetic Minority Oversampling Technique, follows a similar approach to a KNN (k-Nearest Neighbor). It randomly selects a sample in the minority class and looks for k nearest neighbors. One of these neighbors is randomly chosen and a new sample is synthesized between these two points [2]. The 'imblearn' library [3] in python has a module to implement SMOTE. Using this, we include SMOTE in our model pipeline. There are two ways to do this. We can either split our data into training and testing datasets, apply SMOTE on the training dataset and pass it to the model, or include SMOTE in the model pipeline. However, the former is not a recommended way of using SMOTE or any oversampling/undersampling techniques from the 'imblearn' package since this might result in an inaccurate cross-validation score [4]. Hence, we include SMOTE as part of our model pipeline. Among all the models we have trained our dataset on, Decision Tree classifier and XGBoost seemed to have the highest performance. Hence, we choose these classifiers as candidates for us to apply SMOTE on. The performance comparison between these both with SMOTE can be found in Figure 4. The value of k is being changed from 1 to 7.

Models	XGBoost w/SMOTE				DecisionTree w/SMOTE			
Class	0	1	2	Overall	0	1	2	Overall
k=1	0.68	0.82	0.91	0.82	0.63	0.76	0.84	0.75
k=2	0.72	0.78	0.91	0.8	0.63	0.78	0.87	0.77
k=3	0.63	0.8	0.9	0.81	0.69	0.77	0.87	0.75
k=4	0.67	0.79	0.89	0.8	0.71	0.76	0.84	0.75
k=5	0.68	0.78	0.91	0.79	0.69	0.75	0.86	0.74
k=6	0.63	0.8	0.89	0.8	0.62	0.77	0.88	0.76
k=7	0.66	0.8	0.92	0.81	0.64	0.77	0.82	0.77

Figure 4: Comparison between XGBoost and Decision Tree classifiers after applying SMOTE. Classes 0,1 and 2 represent Offensive, Hate speech and Neither respectively. The figure shows the individual accuracy for each class as well as the overall accuracy.

6.10.4 Shortcomings of SMOTE. Along with increased minority class accuracy, SMOTE also comes with certain problems. Due to synthesis of artificial data points, the class separation in the dataset is sometimes heavily affected. To handle this, we see whether SMOTE combined with an undersampling technique can help us overcome this. Two of the most common SMOTE combined with undersampling techniques are SMOTE-with-Tomek5 and SMOTE-with-ENN. In the former, Tomek-links, an undersampling technique is used in removing the samples close to boundaries of classes to increase the separation between the classes [5]. Meanwhile, ENN (Edited Nearest Neighbor) is an undersampling technique which removes instances on the boundary of the majority class where predictions made by the KNN are different from the other majority class points [6]. Both of these methods were originally proposed with binary class data. There are very few studies which support the robustness of these techniques on multi-class datasets. However, we implement this on our multi-class classification to see how it performs compared to our other implementations. This time, we implement these two methods setting the SMOTE parameters to their default values. The results can be found in Figure 5.

Method	XGBoost w/SMOTE-Tomek			XGBoost w/SMOTE-ENN		
Class	0	1	2	0	1	2
Class accuracy	0.92	0.79	0.97	0.97	0.65	0.99
Overall accuracy	0.89			0.97		

Figure 5: Comparison between XGBoost after applying SMOTE-Tomek and SMOTE-ENN. Classes 0,1 and 2 represent Offensive, Hate speech and Neither respectively. The figure shows the individual accuracy for each class as well as the overall accuracy.

7 EVALUATION AND RESULTS

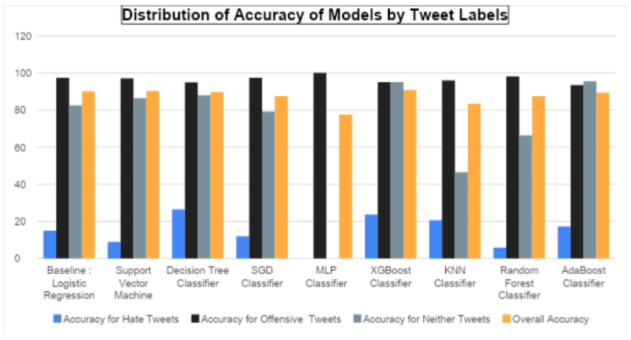


Figure 6: Distribution of Accuracy of Models by Tweet Labels

8 DISCUSSION

From all our observations, we can conclude that improved minority class accuracy comes at the cost of majority class accuracy and in most cases, the overall accuracy too. While SMOTE when combined with Tomek Links and ENN did give very high minority class accuracy and overall accuracy, when we delved deeper into the re-sampling strategy, it revealed the shortcomings of the ENN method on multi-class data. None of the sampling strategies brought a balance in the dataset when using ENN. In Figure 7, we can see the distribution of the data when the sampling strategy was set to 'all' i.e. apply re-sampling on all the classes. However, with all the over-sampling and undersampling, the models developed out of these techniques are highly susceptible to overfitting.

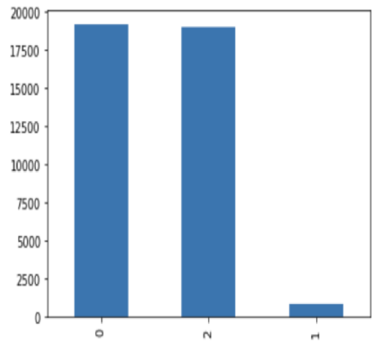


Figure 7: The horizontal axis represent classes 0,1 and 2 i.e. Offensive, Hate and Neither respectively. Note how the majority class - class 1 have become the minority class.

9 CONCLUSION

In this project, we have shown necessary data pre-processing and model building techniques for hate speech detection in social media. Given how similar offensive and hate tweets are both semantic and feature wise, we developed suitable models to classify them. We have also explored a major issue in machine learning tasks i.e. imbalanced data classification and experimented our imbalanced dataset with multiple oversampling and undersampling methods. While very few of these techniques have shown promising results, they do help us remain optimistic about further research in this area like ADASYN.

10 FUTURE WORK

While the majority of the research on imbalance classification focuses on binary classification, it is important to study their impact on multi-class classifications. Class separation being a major concern when oversampling and undersampling, better methods to mitigate this will prevent overfitting and help researchers deploy imbalanced classification models in real world applications.

11 SOCIALLY RESPONSIBLE AI

In our project, we have seen how state-of-the-art models tend to have a bias towards the majority class in case of an imbalanced data. Due to this reason, it is important for AI engineers to handle this imbalance appropriately when developing a model. One should also be cautious when drawing conclusions based on models trained on imbalanced data. This is especially important in case data on religious and racial minorities.

12 REFERENCES

[1] sklearn Documentation : sklearn.ensemble.RandomForestClassifier. Retrieved from: https://scikit-learn.org/

[2] Chawla, N. V., Bowyer, K. W., Hall, L. O., Kegelmeyer W. P.: SMOTE: Synthetic Minority Over-Sampling Technique. Journal of Artificial Intelligence Research 16 (2002) 321-357

[3] imbalanced-learn Documentation. Retrieved from : https://imbalanced-learn.org/stable/

[4] KSV Muralidhar.2021.The right way of using SMOTE with Cross-validation. Retrieved from: https://towardsdatascience.com/the-right-way-of-using-smote-with-cross-validation-92a8d09d00c7

[5] Swana EF, Doorsamy W, Bokoro P. Tomek Link and SMOTE Approaches for Machine Fault Classification with an Imbalanced Dataset. Sensors (Basel). 2022 Apr 23;22(9):3246. doi: 10.3390/s22093246. PMID: 35590937; PMCID: PMC9099503.

[6] D. Wilson, Asymptotic" Properties of Nearest Neighbor Rules Using Edited Data," In IEEE Transactions on Systems, Man, and Cybernetrics, vol. 2 (3), pp. 408-421, 1972.

Received 5 December 2022