Detecting Cyberbullying in Tweets using Classification

1st Sean Ta
Computer Science Department
California State Polytechnic University
Pomona, United States
sta@cpp.edu

2nd Grecia Alvarado
Computer Science Department
California State Polytechnic University
Pomona, United States
greciaa@cpp.edu

3rd Christopher Koepke
Computer Science Department
California State Polytechnic University
Pomona, United States
ckoepke@cpp.edu

4th Abdur Rahman
Computer Science Department
California State Polytechnic University
Pomona, United States
abdurrahman@cpp.edu

***Abstract*—Categorizing posts (known as tweets) as cyberbullying on a popular social media platform called Twitter is exceedingly tedious for human moderators. The relatively high costs associated with training these moderators and the low amount of tweets that are reviewed by them in a given time frame can be vastly improved by the usage of machine learning. We will use Naive-Bayes, Logistic Regression, and Random Forests algorithms that will be trained on a dataset containing real tweets that are labeled as a type of cyberbullying (age, ethnicity, gender, religion, or other). We will transform our data into numerical values by using scikit-learn functions to find the frequency of certain words and assign weights to them to normalize the data. By designing a highly accurate machine learning model, we hope to lessen instances of cyberbullying by allowing moderators to quickly respond to suspected tweets.**

***Index Terms*—Cyberbullying, Twitter, Social Media, Classification, Machine Learning**

I. Introduction

Cyberbullying has become more prevalent as more people turn to social media for daily interactions and as a source of news. Twitter is a social media platform with a more conversational approach compared to other platforms. This paired with user anonymity has made Twitter a breeding ground for cyberbullying. Because Twitter has a large user-base and is used by people of all ages, anyone is susceptible to becoming a victim of cyberbullying. Even worse, public tweets available for anyone to see introduces the danger of herd mentality - more people become inclined to bully others, especially given the ease to do so anonymously and with virtually no consequence.

In order to understand the urgency of this problem, it's important to be able to detect the prevalence of tweets that may be considered cyberbullying. It is also critical to not just consider how often it occurs, but to learn what kind of language can be classified as cyberbullying. The dataset we will be using contains over 46,000 tweets that are classified as either not cyberbullying, or as a specific category of cyberbullying (age, ethnicity, etc). We will implement three different machine learning algorithms. We will specifically train and test models using Naive-Bayes, Logistic Regression, and Random Forests to detect cyberbullying in tweets and compare the results between the three algorithms chosen. Our team recognizes the importance of tagging instances of cyberbullying quickly and accurately, such that Twitter moderators could respond quickly and take appropriate actions. Without the assistance of ML models, these instances may go unnoticed or take so much time to discover that it could cause undue harm to the recipient of these tweets.

II. Dataset

The dataset that we will be using to train and test the ML models is comprised of 46,017 records, in which contains a tweet and its classification. The tweets have been labeled as a certain type of cyberbullying or not. The dataset divides cyberbullying into the categories of age, ethnicity, gender, religion, or other. So every tweet in the data set is labeled as one of those categories, or as not cyberbullying. Discounting duplicates, all of the classes in the dataset have approximately 7900 tweets that are classified to each of them with the only exception being other cyberbullying, which has around 6200 tweets classified to it. This is only 20% lower than the other classes, so overall the class distribution in the dataset is well balanced. The majority of tweets in the dataset are from English speakers, however some tweets are either written entirely in a different language or contain words from other languages besides English.

III. Preprocessing

In the preprocessing stage we first checked if there were any records in the dataset there were missing values, such as an empty tweet; We found no missing values. We next checked for any duplicate records. We found 1675 duplicates and removed them. For our next step, we looked at all the things that we could remove from tweets that were irrelevant to classification. If a tweet contained any links or emojis,

they were removed. If a tweet contained any hashtags, the hashtags were kept but the initial # symbols were removed, along with all other special characters and symbols. Digits, single characters, underscores, and stop words were also scrubbed from tweets. Finally, the remaining cleaned text in each tweet was converted to lowercase and then every word was lemmatized.

IV. Data Visualization

After preprocessing, we analyzed and procured visualizations using the cleaned data. By producing textual visualizations of our data, we can better understand the trends unique to each of our classes - gender, religion, age, ethnicity, not cyberbullying, and other cyberbullying. Both word clouds and n-grams (uni-grams and bi-grams) were generated for each of these classes.

1) Word Clouds: Word clouds represent the frequency of words. Bigger words indicate a higher frequency. The word clouds were limited to the top 500 most frequent words.

Fig. 1. Word cloud for the gender cyberbullying class.

2) N-grams: N-grams represent the most frequent n-sets of words. Uni-grams would simply be the words that appear most frequently. Bi-grams would be the sets of two words that appear most frequently.

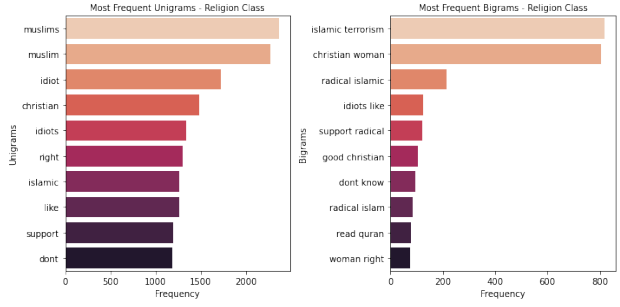

Fig. 2. Most frequent Uni-grams and Bi-grams for the religion cyberbullying class.

V. Methodology

Once our text is cleaned and lemmatized, we imported the 'TfidfVectorizer' function from scikit-learn library to transform our text into numerical values. This function combines the 'CountVectorizer' and 'TfidfTransformer' function together. Applying the 'CountVectorizer' function to the data will count the frequency of each word in the documents. However, if we just rely on the frequency of each word, that means longer documents(tweets) have greater weight. To tackle this issue we used TF-IDF to normalize the data. This stands for term frequency times inverse document frequency and it helps to counter the weight of frequent words. Term frequency looks at the frequency of a word relative to a document. This means the number of times a word appears in a tweet divided by the total number of words in the tweet. Inverse document frequency on the other hand, looks at the inverse of the number of documents that a word appears in. So if a word appears in every tweet in the dataset, the word will have a very low significance. However, if a word appears too infrequently it is difficult for the ML models to utilize them accurately in making predictions. Therefore we set a limit to the number of words that are used. For our dataset, we set the 'TfidfVectorizer' to only use the top 5000 most frequent words. After it calculates word significance in each tweet, it also assigns each unique word in the dataset to a unique number to which it will correspond to. This function essentially transform the text into numerical values, which we can now feed this numerical data into our machine learning models.

The models we planned to use for classifying the text are the Naive-Bayes classifier, Logistic Regression, and Random Forests. We performed multi-class classification to determine if a tweet is not cyberbullying or a certain type of cyberbullying. We began by splitting our dataset into training and test sets with a ratio of 0.67/0.33. This ensures that there is enough data allocated to train our models effectively, and have a sufficient test set to effectively evaluate each model's performance. After splitting the dataset, we trained each model with our training set and then used the test set to evaluate their performance.

We have constructed our first model using a Multinomial Naive-Bayes classifier. The Naive-Bayes algorithm is a popular approach in Natural Language Processing that utilizes the Bayes Theorem to calculate the likelihood of a class for a given piece of text. It calculates the probability of each class for every sample and then outputs the class with the highest probability. For multi-class classification, we calculate the probability of each class (age, ethnicity, gender, religion, other, not cyberbullying) for each tweet, and the class with the highest probability is chosen as the output.

The second model we implemented was Logistic Regression. Logistic Regression is a supervised classification algorithm that finds the probability of a label by calculating the weight of a feature (a word in our case). For each iteration, the weights of these features are updated until they stabilize.

The third model we implemented is the Random Forests

classifier. The Random Forests algorithm consists of a large number of decision trees that operate as an ensemble. Each decision tree produces a prediction, and these predictions are then aggregated. The class with the most votes in the aggregation step is selected as the final prediction for that tweet.

We will compare the 3 different models on the performance metrics of accuracy, recall, precision and F1 score. If the model results are not satisfactory, we will use grid search to tune the hyperparameters of each model. The model with the best performing accuracy in detecting cyberbullying will be chosen as the final model for deployment.

VI. Results

After training the ML models, we first started by looking at the confusion matrices of each model on their testing data.

1) Multinomial Naive-Bayes: We used the default parameters of the MultinomialNB model from scikit-learn to train on our cleaned dataset. The default parameters were: no additive smoothing, and to learn a class' prior probability to fit the data. The following confusion matrix shows the resulting predictions when using our test set on our trained model.

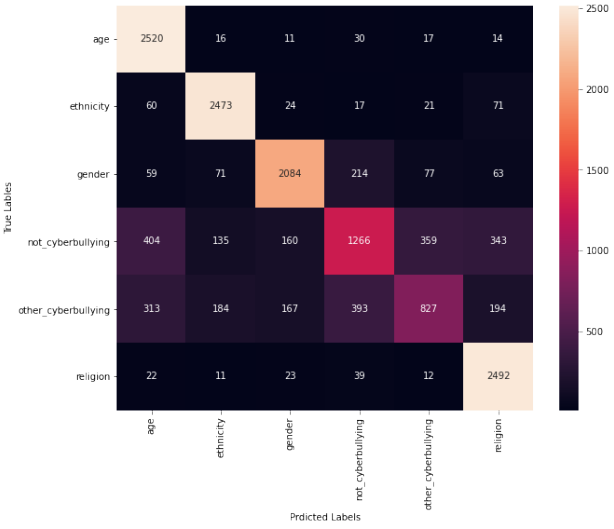

Fig. 3. Confusion matrix for Multinomial Naive Bayes Classifer

2) Logistic Regression: We used scikit-learn LogisticRegression model using the default parameters to train on our cleaned dataset. The default parameters were: use L2 penalty, no dual or primal formulation, use of a constant intercept of 1.0 in the decision function, use of lbfgs solver, no class weight, and no random state value. The following confusion matrix shows the resulting predictions when using our test set on our trained model.

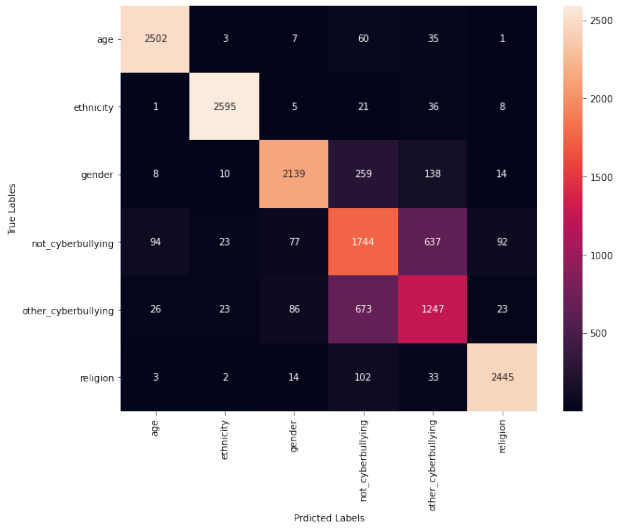

Fig. 4. Confusion matrix for Logistic Regression Classifier

3) Random Forest: We used scikit-learn RandomForestClassifier model using the default parameters to train this on our cleaned dataset. The default parameters were: 100 trees, Gini impurity, no max depth unless all leaves are pure or until all leaves contain less than 2 samples to split, no minimum weight fraction for a leaf node, use of the square-root function to calculate the maximum number of features to use for the best split, no minimum impurity for a split, use of bootstrapping, non-default random state of 50, and lastly no Minimal Cost-Complexity Pruning, warm start, or class weight is used. The following confusion matrix shows the resulting predictions when using our test set on our trained model.

Fig. 5. Confusion matrix for Random Forest Classifier

All three of our models using the described default parameters achieved an accuracy greater than 0.75; specifically, Multinomial Naive-Bayes had an accuracy value of 0.77, Logistic Regression at 0.83, and Random Forest at 0.84. The

weighted average of precision and recall of all the classes for each model had nearly the same values as the accuracy for each respective model. As seen in each confusion matrix, the four highest F1-scores for each model pertained to the label of age, ethnicity, gender and religion. The F1-scores for other and not-cyberbullying labels were much lower, indicating that it is more difficult to correctly predict those labels. During our training and testing phase for each model, we did perform a grid search of various hyperparameters but did not achieve a noticeably higher accuracy, precision, recall or F1 score; Thus for our final models, we used the default parameters of the base models. We printed out the classification report for each model to evaluate the precision, recall, and f1-score. The following tables are classification report for each model.

Classification Report for Naive Bayes:

	precision	recall	f1-score	support
age	0.75	0.97	0.84	2608
ethnicity	0.86	0.93	0.89	2666
gender	0.84	0.81	0.83	2568
not_cyberbullying	0.65	0.47	0.55	2667
other_cyberbullying	0.63	0.40	0.49	2078
religion	0.78	0.96	0.86	2599
accuracy			0.77	15186
macro avg	0.75	0.76	0.74	15186
weighted avg	0.76	0.77	0.75	15186

Fig. 6. Multinomial Naive-Bayes Report

Classification Report for Logistic Regression:

	precision	recall	f1-score	support
age	0.95	0.96	0.95	2608
ethnicity	0.98	0.97	0.98	2666
gender	0.92	0.83	0.87	2568
not_cyberbullying	0.61	0.65	0.63	2667
other_cyberbullying	0.59	0.60	0.59	2078
religion	0.95	0.94	0.94	2599
accuracy			0.83	15186
macro avg	0.83	0.83	0.83	15186
weighted avg	0.84	0.83	0.84	15186

Fig. 7. Logistic Regression Report

Classification Report for Random Forest:

	precision	recall	f1-score	support
age	0.98	0.97	0.97	2608
ethnicity	0.98	0.98	0.98	2666
gender	0.91	0.85	0.88	2568
not_cyberbullying	0.63	0.64	0.64	2667
other_cyberbullying	0.59	0.62	0.61	2078
religion	0.94	0.96	0.95	2599
accuracy			0.84	15186
macro avg	0.84	0.84	0.84	15186
weighted avg	0.85	0.84	0.85	15186

Fig. 8. Random Forest Report

VII. Related Work

Using Machine Learning in Disaster Tweets Classification. Machine learning algorithms have been used to classify tweets as to whether they are related to a natural disaster or not. This proves to be useful in the event where real-time updates on Twitter about a natural disaster can help rescue-teams decide an appropriate course of action. Graduate student Humaid Alhammadi at the Rochester Institute of Technology acquired a data set containing tweets about real disasters and fake disasters.

Preprocessing for disaster classification was similar to our own. This machine learning model cleaned textual data by removing stop words, digits, emojis, and special characters. Data was partitioned into a 80/20 training to test ratio. This is different from our 0.67/0.33 partitioning ratio.

The models used for classifying the text were K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Naïve Bayes, and XGBoost. KNN's accuracy was especially high at 99% and overfitted, while Naive Bayes resulted in the lowest accuracy at 65%. XGBoost and SVM scored 77.9& and 79.6% accuracies, respectively. The Support Vector Machine performed better in regards to true positives when compared to all other models, while XGBoost proved to more accurately find true negatives.

VIII. Conclusion

Our machine learning approach proved to be a highly successful counteractive measure to the growing number of harmful tweets. By successfully interpreting and classifying tweets as a category of cyberbullying, Twitter moderators can more easily detect the types of tweets that require a quick response. All models produced an accuracy greater than 75%. This is a good accuracy considering the models are not used to make any final decisions and are only an assistance to Twitter moderators.

IX. Supplementary Materials

Source-Code
LaTeX Files

References

[1] Shaikh J. Machine Learning, NLP: Text classification using scikit-learn, python and NLTK. Towards Data Science; 2017. [Online]. Available: https://towardsdatascience.com/machine-learning-nlp-text-classification-using-scikit-learn-python-and-nltk-c52b92a7c73a. [Accessed: Oct. 19, 2022]

[2] Alhammadi, Humaid. "Using Machine Learning in Disaster Tweets Classification." RIT Scholar Works — Rochester Institute of Technology Research, Apr. 2022, scholarworks.rit.edu/cgi/viewcontent.cgi?article=12294&context=theses.