STOR565 Project Proposal:

Enhancing Online Safety with Abusive Speech Detection

Group: MLFs

Members: Sathvik Chatta, Lillian Hurban, Elizabeth Singletary, Grace Sosa, Rujula Yete

Link To Our Google Colab Workspace:

https://colab.research.google.com/drive/1PyMW0bTTt7pzj40FGGhQcbY7VBHInhhg?usp=sharing

Abusive Speech Detection

8th March 2024

1. MOTIVATIONS

In our modern digitized world, the need for technological literacy means that children are being exposed to social media platforms and their contents at increasingly younger ages. Despite most popular social media platforms having age requirements to prevent exposing young children to harmful and offensive material, these restrictions are usually about as thorough as the user entering their birthday and are easily and regularly bypassed.

The primary motivation for this project is to use Machine Learning techniques to develop a model that can effectively classify textual data into two categories: abusive language and non-abusive language. Practically every social media platform has textual information that its young users can be exposed to — either text is the primary method through which users interact with the platform (think Twitter, Reddit, any forum board site) or it accounts for a major portion of the site's usage. YouTube, TikTok, Instagram, etc — these are typically thought of as image or video-based platforms, but comments sections, messaging systems, and other features you might see are largely text-based and an integral part of any child's social media user's experience, so the risk of exposure to text content that could be damaging is quite high.

2. GOAL

A model that could successfully classify language as abusive and non-abusive would provide a framework that could be used to carve out more kid-friendly spaces online. With the undoubtedly massive amount of textual data on any given social media platform, human moderators can only look through so much of it. We hope our model — which would automate the process — would assist anyone trying to help younger users safely and easily use their platform. In the future, an example use case could be using our model to build a Chrome extension that acts as an abusive speech filter.

3. DATASET

3.1 Dataset Source & Context

We obtained the dataset, 'hate_speech_offensive'. from HuggingFace.co., which has 24,783 entries. This dataset was originally compiled by Cornell University researchers[1], where they began with a hate speech lexicon containing words and phrases identified by internet users as hate speech, compiled by Hatebase.org, as discussed in their research publication, "Automated Hate Speech Detection and the Problem of Offensive Language." They utilized the Twitter API to search for tweets containing terms from the lexicon, which resulted in a sample of tweets from 33,458 Twitter users, and from there took a random sample of 25k tweets. These tweets were then manually coded by CrowdFlower workers, where each tweet was labeled into three categories, which match its respective 'class' number in the dataset:

Class

- 0: Hate speech
- 1: Offensive but not hate speech
- 2: Neither offensive nor hate speech

```
count                       int64
hate_speech_count           int64
offensive_language_count    int64
neither_count               int64
class                       int64
tweet                      object
```

This dataset is also publicly available on Kaggle. Above is also a glimpse of the variables in the original dataset.

3.2 Data Cleaning & Preprocessing

In a Natural Language Processing project, text cleaning and parsing are crucial for accurate analysis. Lowercasing letters, removing punctuation, and removing stopwords, are just a few things recommended in cleaning strings in NLP (Rastogi). In considering these, these are the considerations we included in cleaning our dataset:

- Lowercased all letters
- Removed URLs/links
- Removed Twitter user handles (any contiguous string after an @ symbol)
- Removed stopwords ('the, 'is', 'and', etc..)
- Removed special characters
- Removed 'rt'
- Removed extra whitespace/spaces
- Created variable, 'tweet_length'
- Original tweets stored in 'og_tweet' column
- Cleaned tweets stored in 'clean_tweet' column

```
count                       int64
hate_speech_count           int64
offensive_language_count    int64
neither_count               int64
class                       int64
og_tweet                   object
clean_tweet                object
tweet_length                int64
```

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

A few things to note, are there were a few exceptions to removing special characters. For example, we decided to keep apostrophes, since the use of certain contractions that may have potential negative connotations like ‘don’t’ or ‘can’t, could be useful for analysis. In addition to cleaning this within a call of one function, we stored the cleaned text, in a new variable ‘clean_tweet’, keeping the unclean tweets, ‘tweet’, to refer back to for analysis.

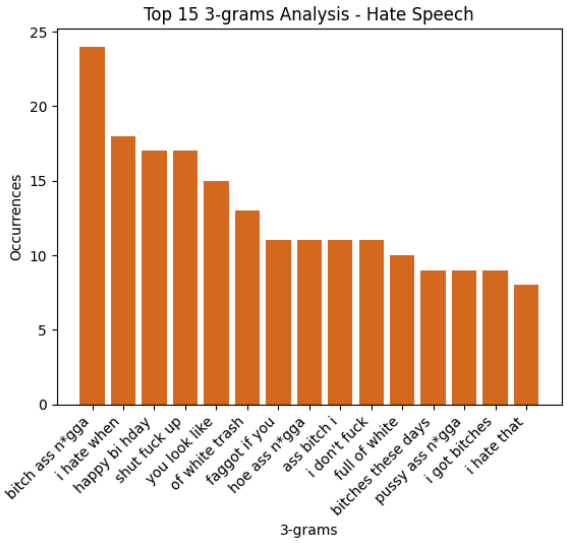
In this context, although we currently are only using Twitter data for analysis, there are future plans to potentially source datasets from other social media platforms, such as YouTube or Instagram comments, as the need for abusive speech detection or abusive speech filters is typically broader than just across Twitter.

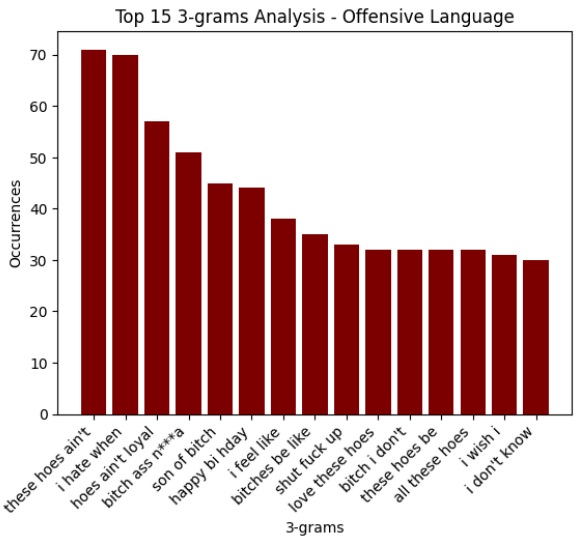
4. EXPLORATORY DATA ANALYSIS

In aiming to explore trends, we explored potential influences of text length, and common word groupings using N-Grams Analysis, and Sentiment Analysis, across the three classes of tweets: Hate Speech, Offensive Language, and Neither.

4.1 N-Gram Analysis

When working with natural language data, several analysis methods can help quantify variations between the classes. One popular method is called N-gram analysis. This counts character or word combinations of a certain length (N) and determines which combinations are most prevalent in the dataset. During the cleaning stage, we separated the Hate Speech Tweets and the Offensive Language Tweets into their own datasets. The N-gram functions from the Natural Language ToolKit (nltk) library were used. After conducting an N-grams analysis on each for combinations of 3 words, we can see that contractions appear to be the most popular combinations in both sets of Tweets.

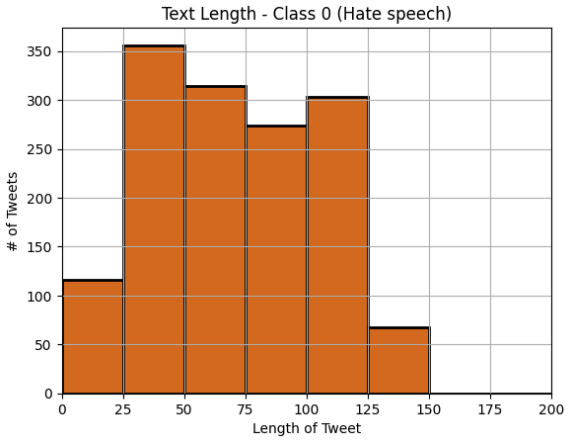


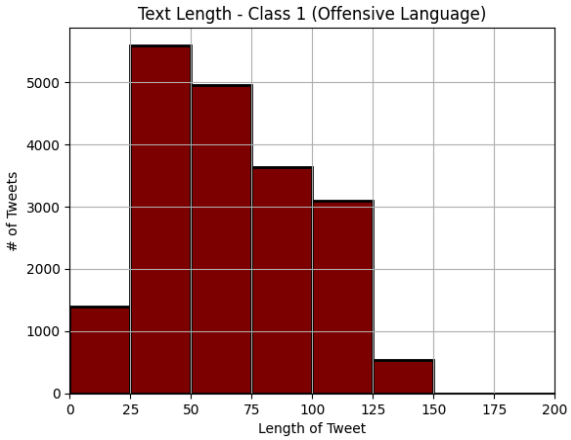


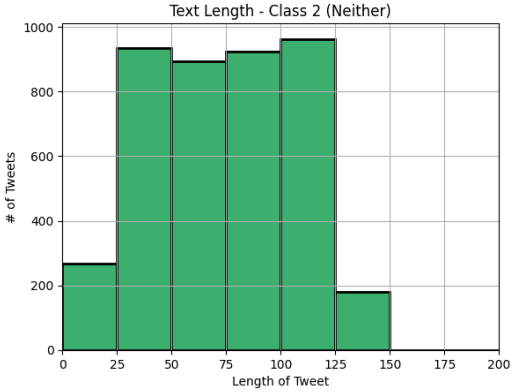
4.2 Tweet Length Analysis

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

Another technique for language analysis is examining the length of the Tweets between the 3 different classes. Histograms were compiled showing the number of characters in each Tweet per category, showing that Offensive Language Tweets may be shorter overall than the other two classes.



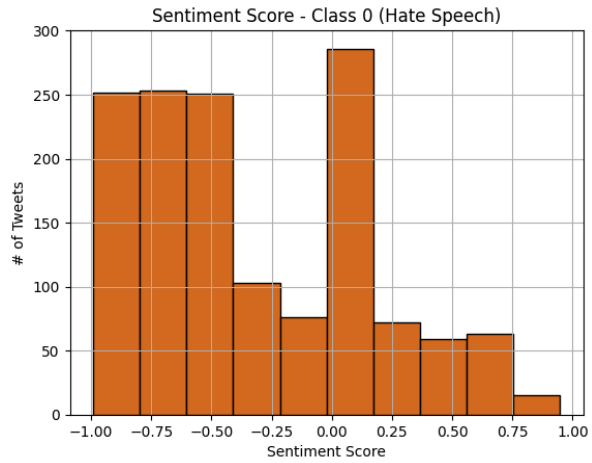


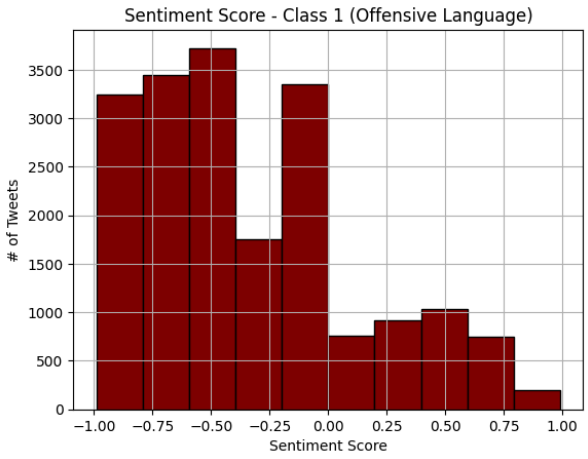


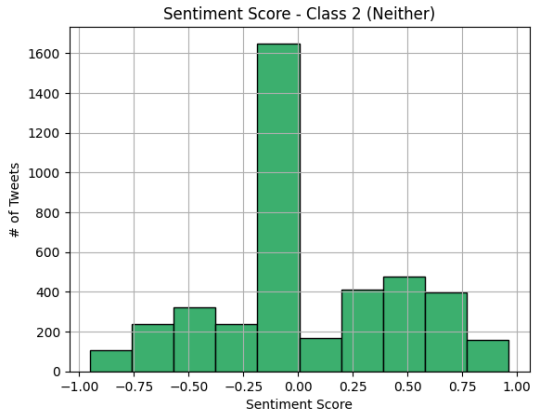
4.3 Sentiment Analysis

Finally, a third method of examining language is sentiment analysis. Based on predetermined classifications from nltk's sentiment analysis tools, scores can be assigned to texts indicating how positive, negative, or neutral the tone of the text is. In this specific analysis, we used the polarity scores, which are indicative of how strong the sentiment of the text is. A larger positive polarity score indicates a stronger positive sentiment, while a lower negative polarity score indicates a stronger negative sentiment. We compared the sentiment distributions of the Tweets between classes. As seen in the histograms, the Tweets classified as neither are mostly concentrated around neutral polarity scores (close to 0), while the Hate Speech and Offensive Language classes are more skewed right (more negative polarity scores). Since this difference is apparent, sentiment score could potentially be used as a factor when training the overall classification model.

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.







5. POTENTIAL MACHINE LEARNING TECHNIQUES

5.1 Support Vector Machine

Since SVMs are effective for high-dimensional data, the use of Support Vector Machines can be suitable for text classification tasks where feature space may be large, due to unique words and combinations of words. Since SVMs can also handle data that is not linearly separable, they can work well with handling non-linear decision boundaries with various kernel functions.

In the context of classifying tweets as abusive or non-abusive, SVMs can learn a decision boundary that separates abusive from non-abusive tweets. Additionally, SVMs have high capabilities to handle binary classification and provide good generalization performance, where we aim to use predictive analysis to classify a variable ‘abusive’ = 0, or ‘abusive’ = 1.

5.2 Logistic Regression

Utilizing features, like the hate speech count, offensive language count, and text length, a simple logistic regression model can help to model the probability of a tweet being abusive, which can

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

be further used for binary classification if the probability is over a certain threshold. To build a more robust model, we have plans to mutate additional variables and utilize feature engineering from further EDA. Additionally, Logistic Regression can handle both linear and non-linear relationships between features and the target variable. Hyperparameter tuning, as well as Gradient Boosting, can help to refine this model and optimize the weight of certain features that contribute to a text's likeliness of being flagged as abusive.

5.3 TF-IDF Vectorizer

The methodology we have been focusing on is utilizing a TF-IDF vectorizer to assign numerical weights to each word in a tweet for every tweet in the dataset. We wanted to steer away from using a counter because common words like pronouns or events might be present in tweets flagged as hate speech but in actuality are not related. Instead we needed a method that would compare a word in a tweet to the rest of the corpus and decide its significance. In the simplest terms a TF-IDF score is calculated by dividing the frequency of the term in the tweet by the amount of times it is present in any other tweet in the dataset. Values close to zero will indicate that the term is common across all tweets and therefore has a lower relevancy. Words such as "the, her, can't, all know) are assigned a low score. This circumvents the issue we found with running an n-gram analysis. Although these words are common in words flagged as hate speech we can effectively ignore them by manually assigning a threshold to the TF-IDF or by running a logistic regression model which also would scale down their importance. We built our TF-IDF vectorizer using the scikit-learn library.

```
#Create Vectorizer
only_text = raw_text['clean_tweet']
vectorizer = TfidfVectorizer()
tfidf = vectorizer.fit_transform(only_text)
tfidf = tfidf.toarray()
tfidf.shape
```

The dimension of our tf-idf matrix is (24783, 33233). Each row is a tweet and each column is a word. Our word bag is generated by the TfidfVectorizer and can be accessed using a built-in method.

```
word_bag = vectorizer.get_feature_names_out()
```

We have 33,233 unique words in the entire dataset.

Next we began building a default logistic model. A model that we haven't fine tuned yet with gradient boosting or hyperparameter tuning.

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

```
#Build a Logistic Regression Model
raw_text['hate_speech_binary'] = (raw_text['hate_speech_count'] >
0).astype(int)
Y = raw_text['hate_speech_binary']
X = tfidf
x_train, x_test, y_train, y_test = train_test_split(X, Y)
model = LogisticRegression().fit(x_train, y_train)
model.predict_proba(x_test)
```

Using scikit's build it method for model evaluation (.score(X_test, Y_test). We were able to obtain an R-squared of 81.89 which is a great starting point. To summarize this core methodology so far, we were able to convert the individual words in each tweet into a numerical value and treat it as a **feature.**

5.4 Recurrent Neural Networks

A Recurrent Neural Network (RNN) is a type of neural network designed to handle sequential data by maintaining a memory state that captures information about previous inputs. This makes them well-suited for tasks like speech recognition, language modeling, and sentiment analysis, where the order of the input data matters.

In the context of analyzing tweets for hate speech, an RNN can be trained on a dataset of labeled tweets to learn patterns and relationships in the text data. The RNN would start with Input Encoding and Sequence Processing, methodologies used to build the algorithm using the semantic relationships between words and the position of the word in the tweet. RNNs then build the classification layer, in this case a binary variable to determine if new input strings are hate speech or not. After that, the RNN would train on the dataset of labeled hate speech we have, to improve the weights of the RNN. While neural networks in general are computationally intensive, we believe that RNNs are effective for this task because they can capture contextual dependencies between words that are spread out over the length of the tweet.

Other potential methods to explore are Random Forests, Bayesian Networks, and other Ensemble Methods.

Conclusion

In conclusion, our project aims to classify text to address the issue of exposure to abusive speech for young users on social media platforms, such as Twitter or YouTube. Through our comprehensive approach that includes data preprocessing, exploratory data analysis, and the application of various machine learning techniques such as Support Vector Machines, Logistic Regression, and Recurrent Neural Networks, we are confident in our model's potential to contribute significantly to the ongoing efforts against online abuse. To refine and build a strong

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

model, we plan to use techniques like Hyperparameter Tuning and boosting. Future expansions could include adapting our model for broader social media platforms and developing user-friendly applications such as a Chrome extension for real-world deployment.

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.

Sources Cited

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). *Automated Hate Speech Detection and the Problem of Offensive Language*. arXiv:1703.04009. Retrieved March 8, 2024, from https://arxiv.org/pdf/1703.04009.pdf

Davidson, T., Warmsley, D., Macy, M., & Weber, I. (n.d.). *Hate speech and offensive language*. Hugging Face. Retrieved March 8, 2024, from https://huggingface.co/datasets/tdavidson/hate_speech_offensive/tree/main

Rastogi, Kashish. (2022, November 22). *Text Cleaning Methods in NLP*. Analytics Vidhya. Retrieved March 8, 2024, from https://www.analyticsvidhya.com/blog/2022/01/text-cleaning-methods-in-nlp/

sklearn.feature_extraction.text.TfidfVectorizer. (2024). Scikit-Learn. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[1]Thomas Davidson, Dana Warmsley, Michael Macy, Ingmar Weber, 2017.