A Machine Learning Approach to Identifying Hate Speech on Social Media

Jarod DeWeese

November 2020

Introduction

The rise of social media has facilitated connections and engagement between people like never before. Unfortunately, these mediums also allow for hateful and offensive language to proliferate, often at the expense of marginalized communities who use such platforms as the main way to connect with those of similar backgrounds. This report focuses on identifying hate speech in social media settings. (Davidson et al. (2017)). Using NLP methods and Machine Learning models, this paper aims to automate this categorization.

I was inspired to choose this project after a university diversity event was disrupted by members of an organization that spammed the meetings with hate speech. A friend of mine was targeted by and deeply impacted by the hateful speech that occurred during the event.

Related Work

To gain familiarity with this issue, several preexisting papers and repositories were reviewed.

In Davidson et al. (2017), the authors explain the legal implications of hateful speech in various parts of the world, as well as explaining how and why certain kinds of hateful speech are more likely to be seen as just offensive, but not hateful. This may exist for any number of reasons, including but not limited to cultural norms, bias, etc. Additionally, they explain that while looking solely for offensive words may initially make it easier to identify hate speech, this naive approach may ultimately be less effective. This is true especially for documents hateful documents that do not explicitly use those terms.

I found Davidson, Bhattacharya, and Weber (2019) which shares 2 authors with Davidson et al. (2017). The paper specifically focuses on the racial bias as it exists in datasets and models. They found that because of this bias, models are more likely to predict that tweets authored by a community are hateful **towards their own community**. As discussed in the paper, this could result in victims being penalized for speaking out, potentially silencing the very communities that are using these platforms to speak out. This phenomenon could be partly explained by the reclamation of words, and the specific vernacular a group may use within their community.

Waseem and Hovy (2016) went into detail explaining why using character n-grams can be useful for this task, especially in conjunction or as a replacement of a word n-gram approach. This paper identified which of these character n-grams were most likely to indicate certain types of offensive speech.

Hate speech is defined as “that it is speech that targets disadvantaged social groups in a manner that is potentially harmful to them” as used in Davidson et al. (2017).

Unsophisticated algorithms may act unfavorably to those groups who are the victims of hate speech. (Davidson, Bhattacharya, and Weber (2019)). This bias is important to acknowledge and work to resolve, particularly in systems that may affect real people.

Of the literature reviewed that focused on automatic classification of this speech online, many authors found that Logistic Regression proved to be consistently effective at determining whether or not a Tweet is hate speech. Specifically, Davidson et al. (2017) used uni, bi, and trigrams of vocabulary and Part of Speech tagged features, as well as counts for URLs, mentions.

Dataset

The dataset used for this project consists of two columns: tweet id and coded value.[1] The original dataset had 80,000 entries, but of those, only the contents of 63,700 entries could be accessed using the Twitter API.

The following is an excerpt from the data: (Founta et al. (2018))

Tweet ID	maj label
848306464892604416	abusive
850010509969465344	normal
850433664890544128	hateful
847529600108421121	abusive
848619867506913282	abusive

Of this data set the data breaks down as follows:

Category	Number	Percent
normal	35998	84.88%
abusive	4530	10.68%
hateful	1881	4.44%

Methodology

1. **Obtain pre-annotated ID dataset** The dataset was downloaded from the source, but this dataset only contained the tweet ID and coding value.

2. **Preprocessing: Obtain actual tweet information**[2] Grab the corresponding tweets, using the Twitter API. Features obtained from this step include Tweet author, author id, text, time the Tweet was authored, if it was in reply to another conversation, and the follower count of the author.

3. **Clean data in the pipeline**
   - Replace mentions, URLs, hashtags
   - Fix encoding irregularities
   - Normalize whitespace
   - Decode emojis
   - Lowercase resulting text

4. **Feature selection** After reviewing existing literature, the following features were extracted.
   1. TFIDF matrix of vocabulary
   2. TFIDF of POS
   3. TFIDF of characters
   4. Count of kind of emojis used if any
   5. Number of users the tweet had mentioned
   6. Flesch reading ease score
   7. Compound Sentiment Score

5. **Adjust for class imbalance** Because there was such a wide class imbalance, a roughly equivalent number of records was chosen from each of the 3 classes to train on.

6. **Models** LinearSVC and LogisticRegression proved to be the most reliable for this application. Each model using roughly 5,000 features per Tweet, after considering the TFIDF matrices features. However, there may be other well-performing models not in the reviewed literature. Although using neural engines were initially considered, the time, knowledge, and resources required to train these proved to not be feasible for the constraints of this project.

7. **Grid search for parameters** After finding the models that were generally most effective, they were optimized using a grid search algorithm, using the total f1 score as the evaluation metric.

Initially, a Jupyter Notebook in Google Colab as the runtime was used for this project. However, this approach soon became limiting with the debugging limitations of the environment, so other platforms

[1] The dataset is available here: https://dataverse.mpi-sws.org/dataset.xhtml?persistentId=doi:10.5072/FK2/ZDTEMN

[2] https://github.com/jdeweese1/cis-531-ml-project

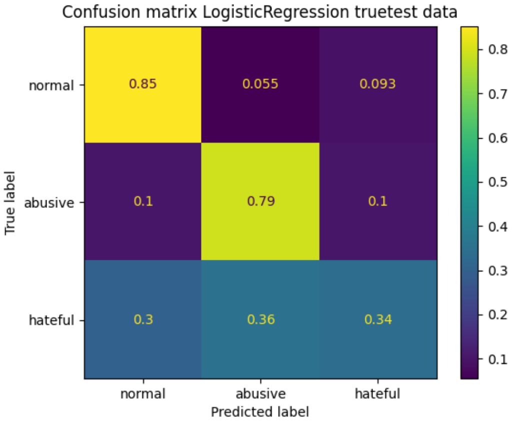
were briefly used. In the end, the code ended up just running as a simple Python program on a local machine, as it allowed for easy examination of the runtime environment, and run specific blocks of code in a non-linear way.

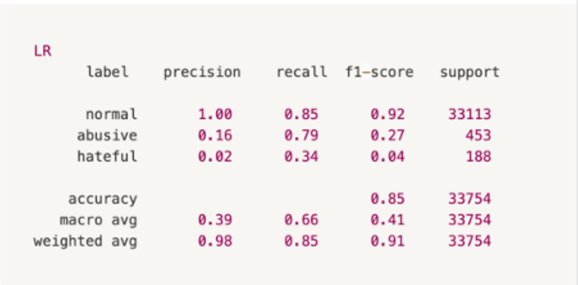
Evaluation

Because there is such a wide class imbalance in the real world with only about 5% of Tweets being hateful, and 10% being abusive it is important that a production model doesn't incorrectly tag normal tweets as abusive or hateful. The baseline for comparison is the metrics mentioned in Davidson et al. (2017): "overall precision 0.91, recall of 0.90, and F1 score of 0.90," "the precision and recall scores for the hate class are 0.44 and 0.61 respectively" This is the baseline to which results will be compared. If the model achieves an overall precision and recall above .8, it shall be considered successful.

Results

Below see the best 3-class confusion matrix, and it's corresponding classification report:



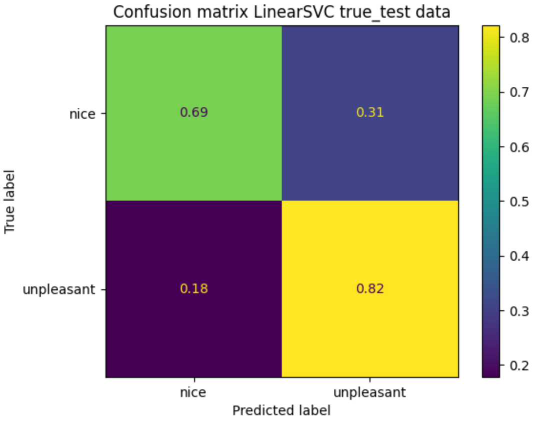


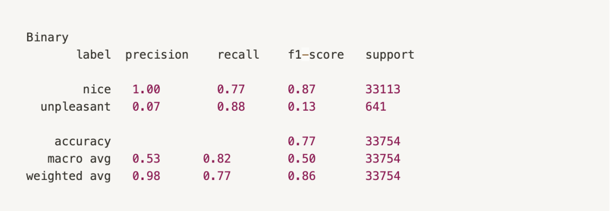
```
LR
      label    precision    recall  f1-score    support

     normal       1.00       0.85      0.92      33113
    abusive       0.16       0.79      0.27        453
    hateful       0.02       0.34      0.04        188

   accuracy                            0.85      33754
  macro avg       0.39       0.66      0.41      33754
weighted avg      0.98       0.85      0.91      33754
```

Below see the best 2-class confusion matrix, and it's corresponding classification report:





```
Binary
       label  precision    recall    f1-score   support

        nice  1.00         0.77      0.87       33113
  unpleasant  0.07         0.88      0.13       641

    accuracy                         0.77       33754
   macro avg  0.53       0.82        0.50       33754
weighted avg  0.98       0.77        0.86       33754
```

As you can see from the figures, the best performing models performed at an overall level consistent with that of Davidson et al. (2017), and on par with the initial goals of this project. However, it is important to point out that even the best model misclassified, 66% of hate speech, 21% of abusive speech, and 15% of normal speech. Somewhat unintuitively, the next best model (2-class) only misclassified 15% of unpleasant speech and misclassified 27% of normal speech. This pattern was consistent for all 3-class models compared with their 2-class siblings.

For multi-class with SVC model, more iterations hurt performance, however for a 2-class dataset, it improved performance, but at the expense of recall for the unpleasant tweets.

Conclusion

While the models showcased can boast respectable f1 scores, they should be interpreted with context. Specifically, because of the wide class imbalance that exists in the dataset (and reality), as long as the model's test data is similarly skewed, and somewhat accurately classifies the normal class, the weighted f1 score gets pulled up with it.

To partially remedy this issue, if the hateful and abusive classes can be consolidated, creating only a 2-class system, model performance can be boosted. However, even in this case, the ratio between nice tweets and unpleasant tweets exists in a ratio of 5 to 1. However, the use case should be carefully considered before deciding to consolidate classes.

Furthermore, because of the reasons discussed in Davidson, Bhattacharya, and Weber (2019), these results should be interpreted with context, and understanding of the inherent bias that exists in data collection, annotation, modeling, and context to cultural norms.

Acknowledgements

- https://gist.github.com/maxogden/97190db73ac19fc6c1d9beee1a6e4fc8#file-paper-md
- https://miki725.com/2019/10/15/markdown-to-pdf-ieee.html
- Xiang (2019) whose code helped to provided useful code samples of how the techniques discussed in the research can be implemented with real model code.
- Emily Davich, who helped to proofread this paper.

References

Davidson, Thomas, Debasmita Bhattacharya, and Ingmar Weber. 2019. "Racial Bias in Hate Speech and Abusive Language Detection Datasets." *CoRR* abs/1905.12516. http://arxiv.org/abs/1905.12516.

Davidson, Thomas, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. "Automated Hate Speech Detection and the Problem of Offensive Language." In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, 512–15. ICWSM '17. Montreal, Canada.

Founta, Antigoni-Maria, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. "Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior." Root. https://doi.org/10.5072/FK2/ZDTEMN.

Waseem, Zeerak, and Dirk Hovy. 2016. "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter." In *Proceedings of the NAACL Student Research Workshop*, 88–93. San Diego, California: Association for Computational Linguistics. http://www.aclweb.org/anthology/N16-2013.

Xiang, Tong. 2019. "Hate Speech Detection for Cosc586." https://github.com/mathfather/Hate-Speech-Detection-For-COSC586.