Team A Final Project

Christian B., Celeste C., Laine B.

Abstract

The goal of this project was to develop an offensive language detection model using a training set of annotated Twitter posts. Our model uses the Scikit-learn TfidfVectorizer [1] class, regular expressions, and comparisons to lists of negative words to extract features from each post. We use the LinearSVC[2] model from Scikit-learn to classify the data as posts that are not insulting, untargeted insults, and targeted insults. The model achieved a macro F1 validation score of 0.4820 on the cross-validation checks.

1 Introduction

This machine learning model classifies social media posts into three categories: not offensive (NOT), untargeted (UNT), and targeted insults (TIN). Posts that do not contain offense or profanity are categorized as not offensive. Posts that contain nontargeted profanity and swearing are categorized as untargeted, and posts that contain insults or a threat directed at a specific individual or group are categorized as targeted insults. We trained the model on a set of 10,592 annotated Twitter posts.

2 Final Model

The final model extracts word counts from the set of annotated posts and transforms the count matrix to a normalized term-frequency times inverse document-frequency representation using the TfidfVectorizer class from Scikit-learn. We include unigrams and bigrams in the count matrix, and we exclude words in Scikit-learn's English stop word list.[3] We also exclude the unigrams and bigrams that appear in more than 45 percent of the posts. The model uses an array of 97,806 features extracted from the training set.

We extract additional features using counts of words contained in publicly available lists of profanities and offensive words (Gabriel, von Ahn). We use additional counts of second person pronouns, special characters, and combinations of these features as additional features in the training set.

The classification model uses the LinearSVC class with a regularization parameter of 10 to classify each post based on the extracted features.

2.1 Validation Measures

When validating the predicted annotations for the training set compared to the actual annotations, the model achieved a macro precision score of 0.9881, accuracy of 0.9933, and a macro f1 score of 0.9844. The model achieved a micro precision score of 0.9933 and a micro f1 score of 0.9933.

2.2 Error Analysis

In the predictions generated for the test set, the model results in a false positive identification of a targeted insult for the following post: “She is a Skrull. Enemy of The Kree. The Kree are who gave Carol her powers and whose uniform she is

[1] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[2] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html?highlight=stop%20words#sklearn.feature_extraction.text.TfidfVectorizer.get_stop_words

wearing in the first few moments of the trailer." This is likely due to the inclusion of "enemy", which the model cannot determine to be related to a science fiction movie rather than directed at another individual.

The model also resulted in a false negative identification as not offensive for the following post: "you are a sick individual. Get a life or make the world a better place or end yours. No loss trash IMHO." This is likely due to the use of "sick" and "trash" as insults, which may occur frequently enough in other contexts that the model is not able to identify them as insults in this context.

3 Alternative Methods Tested

3.1 Scikit-learn Classes

We used the Pipeline [4] and GridSearchCV[5] classes from Scikit-learn to identify optimal parameters for the initial count matrix. The model resulted in lower macro F1 scores when using the CountVectorizer[6] class to generate the count matrix, the SelectKBest[7] class to select included features, the StandardScaler[8] class to standardize the count array, and the RandomForestClassifier[9] and LogisticRegresionCV[10] classes to classify the posts.

We tested the range of inclusion of unigrams, bigrams and trigrams, with and without the English stop word list. We also tested document frequency thresholds from 20 to 100 percent. The final model uses the highest scoring combination of these factors.

3.2 Additional Feature Extraction

We initially included counts of generally positive and negative words in the features added to the count matrix generated by the TfidfVectorizer, which resulted in lower validation scores. We converted emojis into words to include emojis associated with negative words in the additional features. This resulted in lower validation scores as well. Additional features tested that resulted in lower validation scores include a count of capitalized words and a count of user tags.

Acknowledgments

We would like to acknowledge the instruction, encouragement and support of Dr. Anthony Rios, Assistant Professor, Department of Information Systems and Cyber Security at the University of Texas at San Antonio.

References

Luis von Ahn. Offensive/Profane Word List. https://www.cs.cmu.edu/~biglou/resources/

Robert J. Gabriel. Full List of Bad Words and Top Swear Words Banned by Google as they closed it. 2016. https://github.com/RobertJGabriel/Google-profanity-words

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12 (Oct), 2825–2830.

[4] https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

[5] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[6] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

[7] https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

[8] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[9] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[10] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegressionCV.html