Abstract

This project seeks to scrape posts from a digital reddit community and learn patterns from the content and language used to offer recommendations to the end user on how to produce a post to be viewed favorably by as many people within the community as possible. 24,000 reddit posts from r/conservative were used to train a deep neural network to predict a karma score based on features extracted from the post. Once trained, the model is used to evaluate content and evaluate mutations to the post, thus enabling predictions for the optimal posting time and alterations to the specific language used within content to maximize the predicted karma score. The output is visualized to the end user through an interactive GUI.

Introduction

As the 9th most visited website, according to search engine results, Reddit attracts hundreds of thousands of users to the site monthly. The website allows users to subscribe and post to smaller communities of users, often joined together by a specific characteristic, known as “subreddits”. In effect, Reddit enables users to sort and label themselves into categories. This project seeks to use this existing structure to build a dynamic corpus to learn about the types of attitudes and language used by specific subreddits, and thus specific groups of people.

In addition to posting and viewing content, users have the ability to vote on content, the culmination of which is totalled into a single “karma” score. This karma score is important because it directly affects the ranking of posts shown to a new user when they request content from the subreddit. Moreover, the karma score represents a quantifiable metric for the level to which a community “agrees” or “disagrees” with a particular post. This project seeks to use this property as a label to train a deep neural network, thus enabling the model to begin to uncover subtleties in the shared opinions and attitudes of members of the subreddit.

To highlight the potential usefulness for such analysis, r/Conservative, (a highly active subreddit with above 700,000 members) was chosen for analysis. The subreddit attracts users based on a shared, Conservative ideology. Thus, a model capable of predicting how well content will fare to Conservitive users would be invaluable to politicians, advertisement companies, news organizations, campaigns, and public relations personnel. Moreover, the model could identify subtleties of the ideology that may be useful in predicting the behavior and attitudes of Conservative people.

Since the model is trained from Reddit, the model has access to a never-ending stream of data. Thus, as attitudes of Conservatives change on particular issues, the model itself will be able to adjust its predictions to match the community’s attitudes without requiring attention from a machine learning engineer.

Methodology

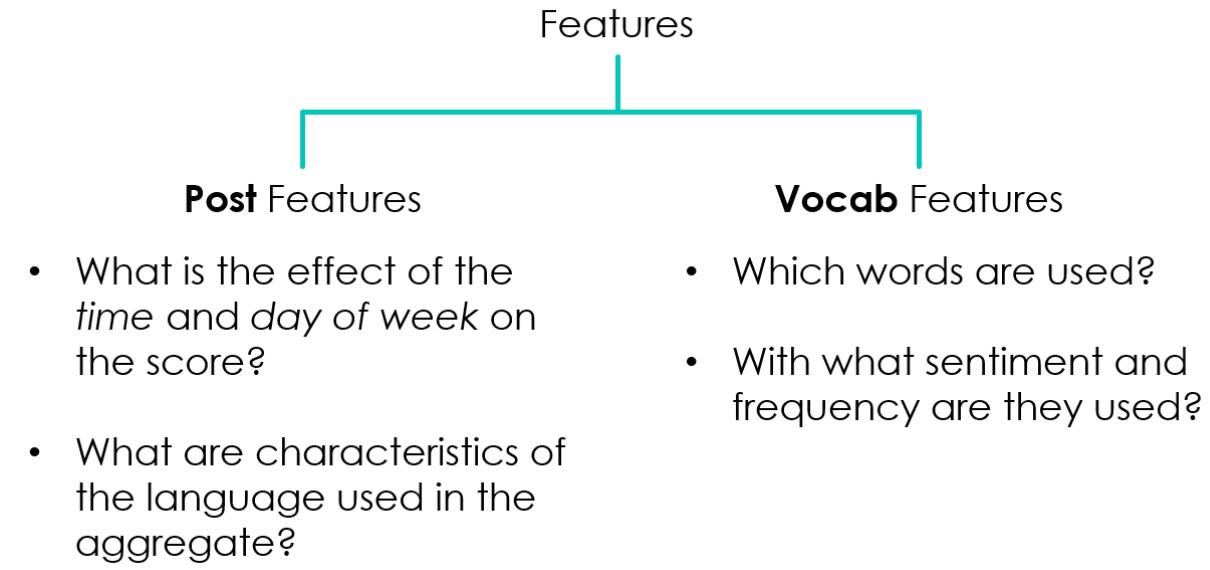
Post Scraping

Reddit posts were scraped through pushshift.io, a free API service that stores all Reddit posts in JSON. Each post contains the language of the post itself as well as attributes of the post, including the time the post was created and the karma score associated with a particular post.

Because karma scores are dynamic and change rapidly as the community votes on new content, posts were only scraped if they were published at least two weeks before the time of scraping. Thus, new posts with low karma scores simply because the community has not had ample time to review the content would not bias the model. Posts were stored into a text-based file using Pickle.

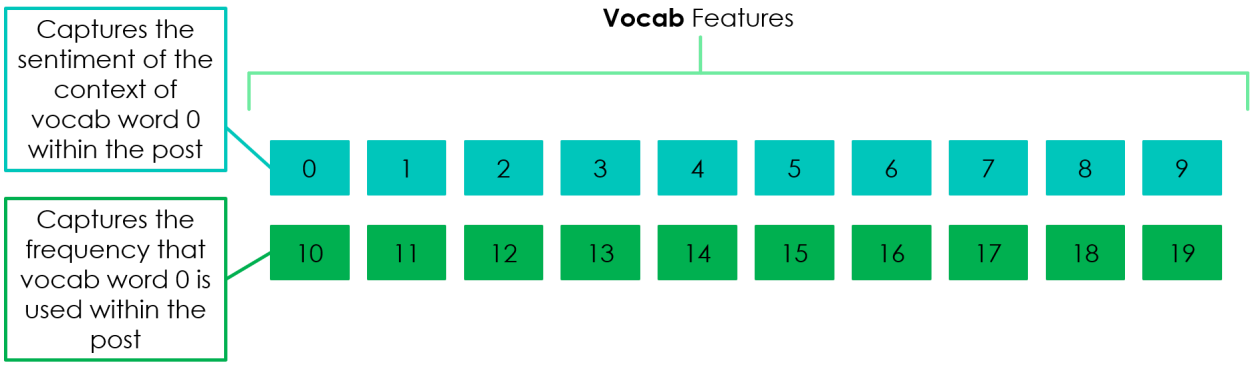
Feature Extraction and Training

For each post, 2 sets of features are generated. Post features describe characteristics of the post as a whole. Vocab features describe the effects of individual words used within a post and attempt to gauge the community's opinions on subjects and combinations of subjects.

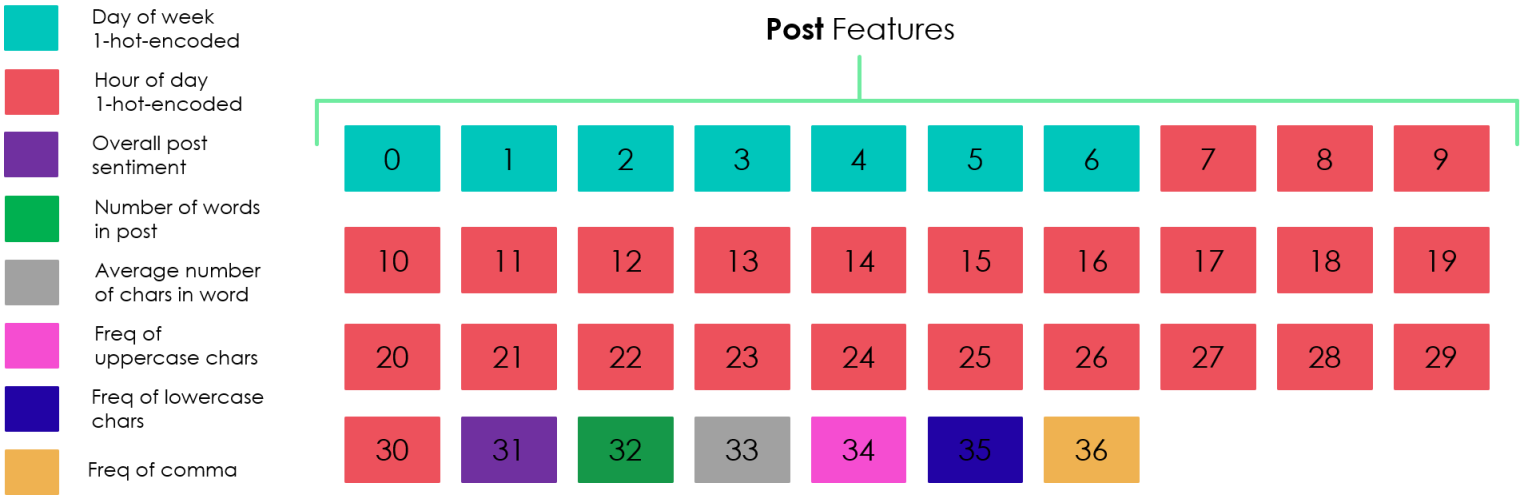


To generate the set of Vocab features, a vocabulary is built consisting of the 1000 most frequently used words in the post corpus. For normalization purposes, stopwords and punctuation tokens are excluded, and each token is lemmatized.

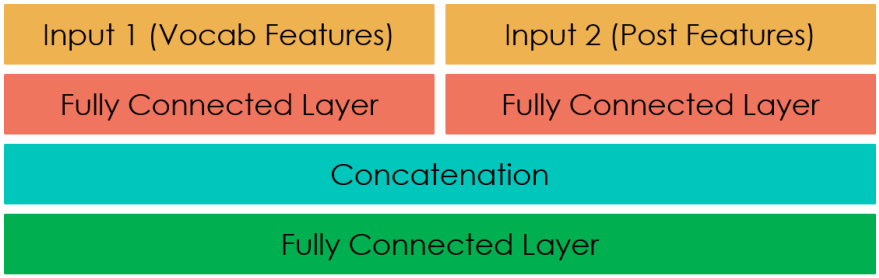
Each post is tokenized into sentences, and the sentimentality of the sentence is computed. The sentence is further tokenized into tokens. If a token matches one from the vocabulary, the sentiment of the sentence in which it is used and the frequency of which the token is used within the post are stored into the Vocab features array according to the index of the vocabulary word.



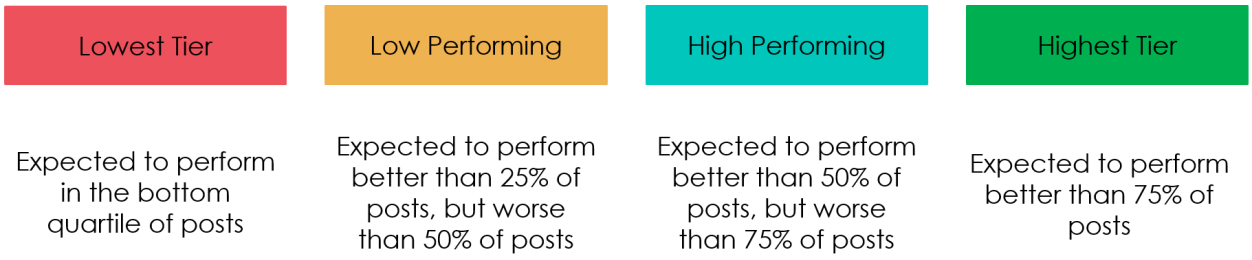
Additionally, each post is assigned a set of 37 post features that capture characteristics of the post in the aggregate. These features account for the time (in UTC) and day of week that a post was published, and for descriptive statistics for the post as a whole, such as the average number of characters in a word. In this capacity, the model can evaluate whether a particular post matches with the community specific norms for language; encompassing word complexity, sentence complexity, case sensitivity, etc.



Features are generated for each post within the training set, and a deep neural network is designed adhering to the following architecture:



Each post is encoded into one of four categories, according to the karma score associated with the particular post.



The model was then trained for 50 epochs.

Computational Analytics

For any given post, the model is capable of producing a prediction for a particular category with a particular confidence score. Thus, for any given post, the question becomes "how can the post be modified to increase the predicted score"?

To answer this question, a custom evaluation is imposed on the final logit scores such that one post is considered better than another if it is predicted to land in a higher category, or if it is predicted to land in the same category with higher confidence. This result allows us to perform mutations to content and measure if the mutation is expected to perform better than the original.

Using this method, for a given input of text the model can predict the best time to post the content by performing predictions on each available date and time configuration in the feature space.

Likewise, for a given input of text, we manipulate the feature space to use the model to perform predictions on how a post would be expected to perform if certain words were removed. These predictions are visualized to the user through the GUI.

Results

Metrics

The model achieved an average accuracy of 75% on the testing data. Since the model places a post into one of four categories, the accuracy score treats all errors the same; that is, a prediction of 0 for a post labeled as 3 affects the average in the same way that a prediction of 2 would for the same post. The model also achieved an AUC score of 0.88.

When reducing the model to binary classification (in which, the bottom two tiers are considered a single class and the top two tiers are considered a single class), the model achieves a custom accuracy of 87% on average. In this sense, the model is quite good at identifying poor content and quality content.

The model takes approximately 10 minutes to train in its entirety on my 4 core machine with 24,000 posts used.

Visualization GUI

To provide meaningful feedback to content creators, information from the model must be delivered in an appealing and interpretable way. As such, the project's GUI describes and visualizes its predictions as well as the computed changes that the model recommends to increase the predicted score for a user supplied post. All post predictions found with the GUI presume that the end user is posting at the time the <<ENTER>> key is pressed.

Conclusion

This relatively simple model achieves an incredibly high accuracy score on such a dynamic, real world corpus. To content producers, the information that the model provides is insightful, valuable, and actionable. To companies and politicians, the insights can translate into success or failure.

Ultimately, I believe that the success of this model asks an existential question: does the accuracy of this model convey more information about the model or the subreddit? As a frequent user of Reddit, it is startling to me that a neural network can differentiate, with 87% accuracy, good content from bad content, and thus explain the behavior of hundreds of thousands of users.

While the project has not demonstrated causation, I believe that the accuracy and success of this project quantifies the effects of a live, functioning echo chamber in a world of total polarization. In fact, I believe that in future studies, the accuracy of this model could be used to measure an existential feature of a community. If a computer application can explain the success of content, then perhaps that measurement can be seen as a statement of the intellectual diversity and complexity of the community.

References

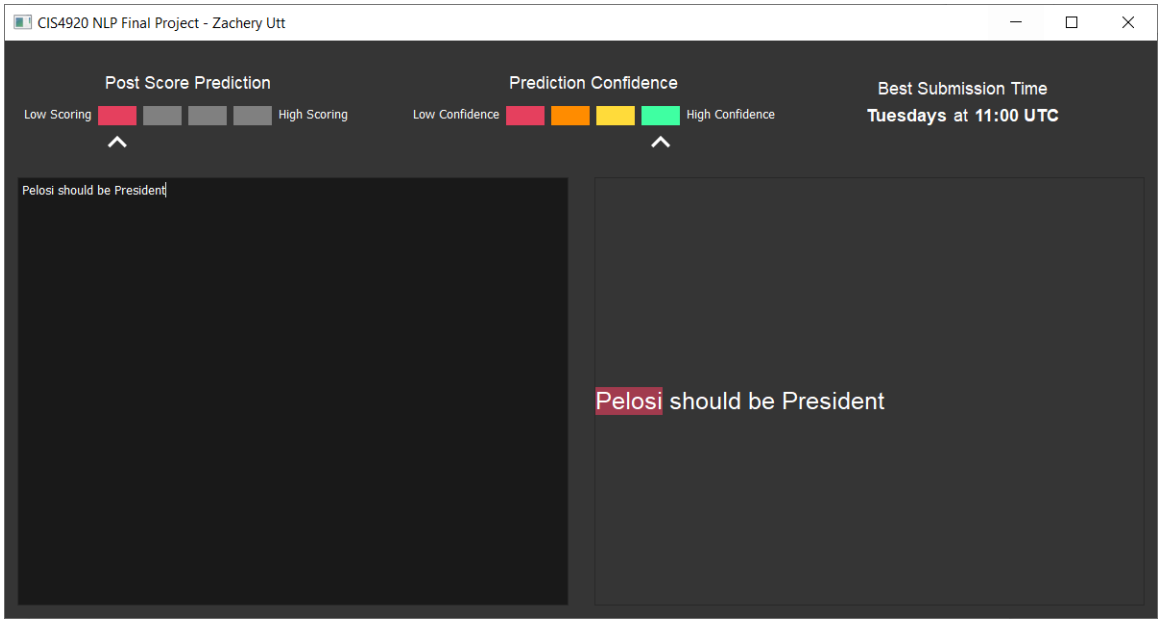
Hardwick, J. (2021, January 26). Top 100 most visited websites by search traffic (2021). Retrieved April 27, 2021, from https://ahrefs.com/blog/most-visited-websites/

Herrmann, M. (2018, January 15). Dark theme for qt widgets? Retrieved April 27, 2021, from https://stackoverflow.com/questions/48256772/dark-theme-for-qt-widgets

Pipis, G. (2020, October 11). How to run sentiment analysis in python using vader: Python-bloggers. Retrieved April 27, 2021, from https://python-bloggers.com/2020/10/how-to-run-sentiment-analysis-in-python-using-vader/

GUI Appendix

Example 1: This post is predicted to perform poorly in a conservative subreddit. The model is highly confident that the provided text will score in the bottom quartile. Moreover, the model has identified that removing the word “Pelosi” would increase the prediction of the model; in other words, the word “Pelosi” in this context partially explains why the model predicted this post to perform so poorly. The user should instead choose a different word.



Example 2: This post is predicted to perform well on the conservative subreddit. The model is highly confident in its assertion, and has identified that the words “Pelosi” and “removed” increases the likelihood that this post will perform well.