Introduction

The goal of this project was to explore, clean and analyze Reddit data (of the subreddit *unpopularopinion*), identify one or more questions of interest based on the data, and use the tools that we have learned in PPOL 567 to produce an interesting result.

A Decision Tree Regressor was used to understand whether certain predictors were important determinants of the score (votes) of a comment, while a Random Forest Classifier was used to determine whether, given certain predictors, we can predict whether the comment would be popular/well-liked.

Code

All code for this project is included in a well-commented Jupyter Notebook, titled 'ms4684_project.ipynb'. This notebook can also be found here.

Data Exploration and Analysis

I started by exploring the *unpopularopinion* subreddit dataset. This included looking at the number of rows and columns, checking the data types, as well as viewing the number of missing values in each variable in the full dataset. A lot of variables had a high number of missing values, sometimes more than their available values. Since variables with a lot of missing values do not add too much information, I removed the variables that had more than 80% of their values missing. This reduced the number of variables by about 12.

As a non-reddit user, a lot of the variables did not make too much sense to me at first. Therefore, I relied on online sources to understand what the variables meant. Based on this, I looked closely at a few important variables. This included looking at unique values of a variable (e.g., controversiality, which only had only values of 0 and 1), and analyzing descriptive statistics of numeric variables (e.g., looking at the score variable allowed me to see that on average, comments on the *unpopularopinion* subreddit receive more upvotes than downvotes). Closely analyzing each variable like this not only allowed me to understand each variable, but also allowed me to create new variables. For instance, I used the 'created_utc' variable (denoting when the comment was posted) to create a new column that only captured the date, and another column to denote whether the comment was posted on a weekend or not.

As part of my data exploration, I also looked at the number of comments posted over time. Since the full dataset was too large, I had to visualize this for just one month's data. The visualization for that month showed that most comments were posted in the beginning of the month, and the least towards the end of the month.

While analyzing the 'total_awards_received' variable (which denotes the number of awards received by the author of the comment), I also decided to look at the relationship between this variable and the score of a comment. The visualization showed that the relationship between the two was poor, and that most comments in the *unpopularopinion* subreddit were posted by

authors with less than 10 awards. The weak relationship between the two variables was confirmed by calculating the actual correlation (0.3). The correlation for all numeric variables in the dataset was also visualized which showed that only two variables ('locked' and 'stickied') had a very strong, positive relationship. None of the other variables had a very strong relationship, either positive or negative.

The comment text ('body' variable) was also pre-processed and converted to lowercase. A binary variable was created to indicate whether the comment text contained one or more of the top 3 nouns of the *unpopularopinion* subreddit (obtained from here). After this, the text was tokenized and a new variable was created that counted the number of words in a comment. This included stop words, since I was interested in the length of the comments, not necessarily the content.

Model 1:

For the first model, I chose the comment score as the dependent variable, and wanted to determine whether the independent variables chosen would be important determinants of the score. The independent variables were selected based on how much the variable made sense and was easy to comprehend, as well as what I thought could possibly have an impact on the score. These independent variables therefore included the total awards received by the author of a comment, the number of gildings (gold stars on a comment), whether the comment was posted on a weekend, whether the comment was controversial, the type of user who posted the comment, the number of words in the comment text and whether it included the top 3 nouns of the subreddit.

The model chosen was a Decision Tree Regressor, because as compared to other models, decision trees require less pre-processing and normalization of data, and are easy to interpret. I only selected the relevant variables from the dataset (all the independent variables and the dependent variable), removed the missing data, and split the data 80:20 (80% training and 20% test), after which I indexed the 'distinguished' variable since it had two string labels (I did this for the train and test set separately to avoid any information from the test set leaking into the training data). Next, I created a feature vector by combining all the features together using the 'vectorAssembler' method. I used this method because it transforms features and feature transformer results to a single feature vector which can then be used to train models. A decision tree regressor was fit on the training data, and predictions were made using the test data. Since a decision tree regressor object does not have a summary object, only the root mean squared error was analyzed. The RSME of 52, although high, is context dependent. In this case, since comment scores have a wide range, an error of 52 is not too bad, although it isn't good either.

Model 2:

In the second model, a Random Forest Classifier was used to determine whether, given certain predictors, we can predict whether the comment would be popular/well-liked.

For this model, I created a binary variable to indicate whether the score was above (labeled 1) or below (labeled 0) the average score. The independent variables were the same as before, and the

model chosen was a Random Forest Classifier because random forests can handle large datasets efficiently and overcome the problem of learning with just a single tree. Once again, I only selected the relevant variables from the dataset (all the chosen independent variables and the dependent variable), removed the missing data, and split the data 80:20 (80% training and 20% test), after which I indexed the 'distinguished' variable. Next, I created a feature vector by combining all the features together using the 'vectorAssembler' method. A random tree classifier was fit on the training data, and predictions were made using the test data. The area under ROC, F1score and accuracy were analyzed in order to evaluate the model's results.

Results

The first model (Decision Tree Regressor) had a Root Mean Squared Error (RSME) of approximately 51. RSME represents the square root of the variance of the residuals, and knowledge of the data is required in order to analyze the figure correctly. Given the wide range of comment scores, an RSME of 51 is not too bad. However, it is not good either, since the lower the RSME, the better. Based on this, the independent variables chosen are not the best determinants of a comment's score.

The second model (Random Forest Classifier) had an area under ROC of 0.6. This shows that the model does have some class separation capacity i.e. predictive power, but it is not very strong, as the closer to 1 the better. In terms of accuracy, the model predicted 58% data points correctly. This is better than chance, but not strong predictive power. Lastly, the F1 score of 0.5 for the model shows that the model is reasonable, and not completely useless (as it would have been if the F1 score was 0). Based on this, the independent variables selected seem to be moderate or slightly above average predictors of whether the score of a comment is going to be above or below average.

Validation

In terms of validation, before running the second model I addressed the imbalanced dataset (with few comments with scores above average), by under sampling the majority class (where comments were below average). Creating a more balanced dataset made the results more reliable, as the final accuracy figure could no longer be attributed only to a simple prediction of the majority class.

Future Work:

I was interested to preprocess the raw comment text data, and use it as an independent variable, but the vectorized sparse matrix was too large and made fitting the model a very lengthy process. I would like to find out a way to still do this.

I am also interested in looking at the sentiment of a comment and seeing the correlation with the score that it receives. Other questions that comment sentiment would allow me to explore are: Are positive comments more likely to be upvoted? Are authors with more awards more likely to write positive comments?

In terms of what I would have done differently if I had more time, I would have used more independent variables. There might be important variables that I might have overlooked given my lack of reddit knowledge and that would have been important for the model. It would be helpful to talk to a domain expert (i.e. a frequent reddit user) to understand the data better.