DSE 220: Final - Predicting Amazon Review Helpfulness

Professor: Volkan Vural

Teaching Assistants: Chetan Gandotra and Tushar Bansal

Joshua Wilson (kaggle name: melvin) A53228518

1 Objective

The task of this project is to predict whether an Amazon user's review of an item will be considered helpful. We are provided 200,000 reviews and associated data to be used for training a prediction model, which will be evaluated against a test set of 14,000 reviews. A brief summary of the various data associated with each review is below:

1. itemID - The ID of the item. This is a hashed product identifier from Amazon.
2. reviewerID - The ID of the reviewer. This is a hashed user identifier from Amazon.
3. helpful - Helpfulness votes for the review. This has two subfields, 'nHelpful' and 'outOf'. The latter is the total number of votes this review received, the former is the number of those that considered the review to be helpful.
4. reviewText - The text of the review.
5. summary - Summary of the review.
6. price - Price of the item.
7. reviewHash - Hash of the review (essentially a unique identifier for the review).
8. unixReviewTime - Time of the review, i.e, time elapsed in seconds since 1970. For example - 1400544000
9. reviewTime - Plain-text representation of the review time in human readable format. For example - '05 20, 2014'
10. category - Category labels of the product being reviewed.
11. rating - Rating given by the reviewer.

The general modeling approach utilized for this project was to engineer features from the provided data, split the data into two groups based on the number of helpfulness votes, and perform an ablation experiment to identify the best subset of features from which to tune regression models.

All code associated with this analysis can be found on GitHub:
`https://github.com/mas-dse/jsw037/tree/master/DSE220`.

2 Data Preprocessing

Upon reading the data from json, we extract the 'helpful' subfields ('nHelpful' and 'outOf' for the training data, and 'outOf' for the test data), and determine the percentage of votes deemed helpful by dividing 'nHelpful' by 'outOf'. Next, we split the 200,000 row training dataset into training (80%) and validation (20%) datasets using a random state of 55. From the training dataset, we then remove outliers with 'outOf' greater than 80 where less than 20% of votes were helpful.

3 Feature Engineering

From the provided review data, we determine the total number of reviews made by each user, and for each item, as well as the average helpfulness rating by user and item. At this point, rows for which 'outOf' is zero are dropped, and we calculate the total number of rated reviews (i.e. 'outOf' > 0) by user and item, and the percentage of total reviews by user and item that were rated.

We then determine the number of categories in which each item appears. Next, we flatten the entire list of categories to identify the 200 most common categories, and create binary variables to indicate which items belong to which of these 200 most common categories.

Many features were derived from the review text and summary, including length both in terms of number of words and number of sentences, and the number of total syllables, which were derived using information from the Carnegie Mellon University Pronouncing Dictionary (see `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`). From the number of words, sentences, and syllables, Flesch-Kincaid readability metrics (see `https://en.wikipedia.org/wiki/Flesch-Kincaid_readability_tests`) were calculated. We also determined the percentage of capital letters, and the percentage of words and sentences that were entirely capitalized. Finally, we calculated the fraction of characters in the review and summary text that were punctuation.

After adding the features described above, missing and null values were imputed by replacing with 0 (as they were generally null due to attempted division by zero). Several columns no longer needed were dropped from the data ('categoryID', 'categories', 'reviewText', 'reviewHash', 'summary', 'helpful', and 'reviewTime'), resulting in 225 feature columns in our final processed training dataset.

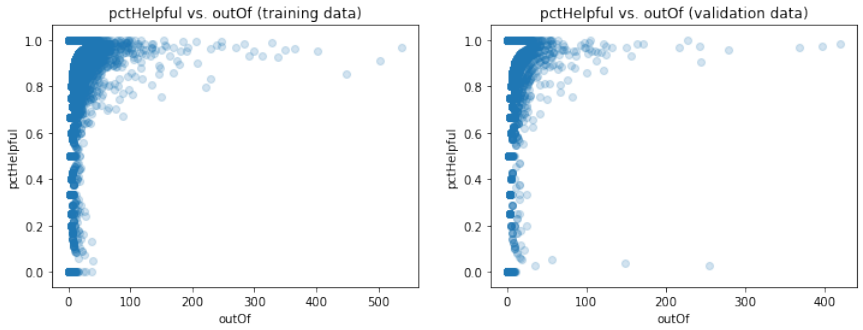
The validation and test datasets were processed similarly, and all processed datasets were written to csv files.

4 Exploratory Data Analysis

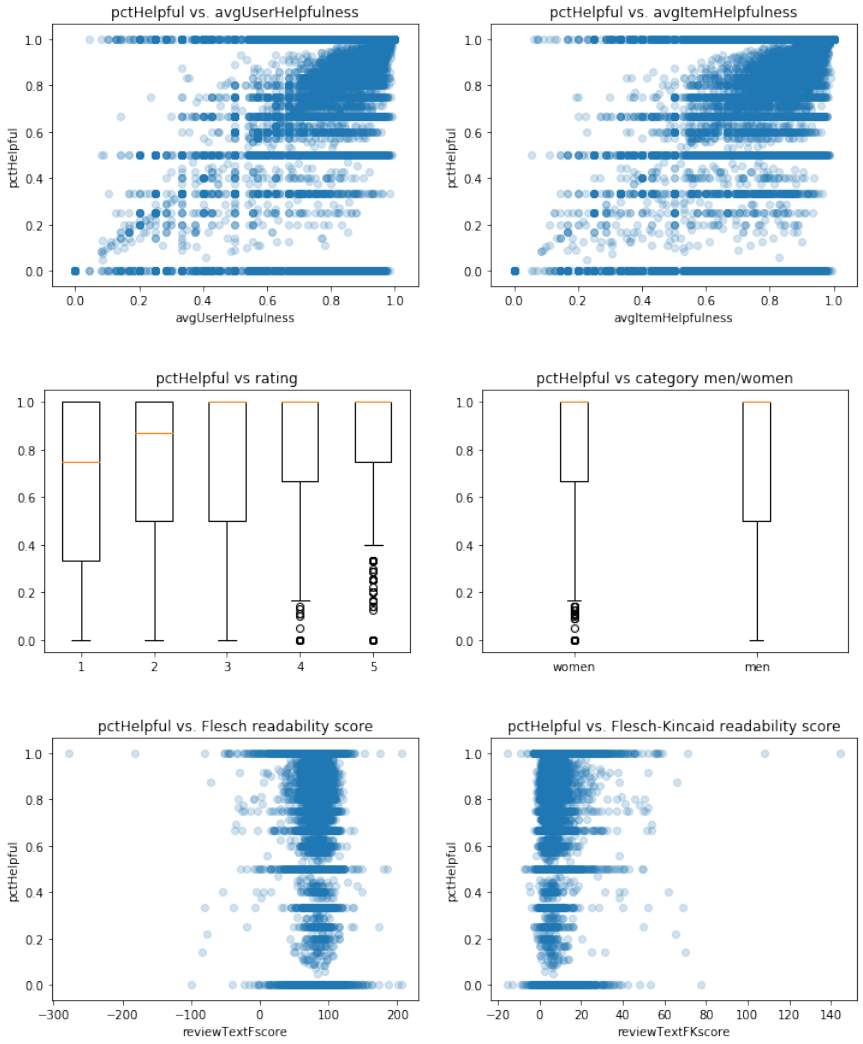
Exploratory analysis was performed in order to understand our data and to identify potential problems, such as outliers. First, we determined the correlation between each feature and our target variable, 'pctHelpful'. The features most strongly correlated to our target variable are displayed below:

avgUserHelpfulness	0.716593
avgItemHelpfulness	0.585750
rating	0.178120
nHelpful	0.140151
outOf	0.081301
category_Women	0.066301
reviewTextNumSents	0.055915
...	
category_Athletic	−0.038399
reviewTextPctSentsAllCaps	−0.040773
reviewTxtPctCapsLetters	−0.043782
reviewTextPctWordsAllCaps	−0.044579
category_Running	−0.048539
unixReviewTime	−0.049237
category_Men	−0.067828

Based on the strength of variable correlation with our target variable, we created a series of plots. Several that are for features in our final model are displayed below:



In the plot of 'pctHelpful' versus 'outOf', we can clearly see at least two outliers in the validation dataset where 'outOf' is very high but 'nHelpful' is not. Recall that these types of outliers were removed from our training data. We can also notice that the distribution of 'outOf' is very skewed, with many more low values and a long tail of high values.



5 Prediction Model

Now, we attempt to build a model to predict 'nHelpful'. We predict 'nHelpful' indirectly, however, as we first predict 'pctHelpful', and multiply our prediction with 'outOf' to obtain a prediction for 'nHelpful'. The general approach was to use linear regression. After trying various models on the full dataset of over 200 features, it was determined that an ElasticNet model fit to the training data with the 'l1_ratio' parameter set to 0.01 and the 'alpha' parameter set to 0.25 performed better than other models on validation data, so all subsequent analysis was performed using ElasticNet with these parameter values.

Due to the long tail for 'outOf' in our data, we split the dataset into two parts. It was determined that Mean Absolute Error (MAE) was minimized on the validation data when separate models were fit to the training data based on an 'outOf' threshold value of 21 (i.e. all data for which 'outOf' was under 21 was used to train one model, and all data for which 'outOf' was 21 or higher was used to train a different model).

Finally, separate ablation experiments were performed on the split datasets. To identify an appropriate subset of features for our final models, we iteratively fit models to the training data after removing individual features, then calculated the MAE of predictions made on the validation dataset, and removed the feature at each step whose removal most decreased MAE. This process was continued until it was no longer possible to improve validation MAE by removing any features. For low values of 'outOf', the ablation process removed 213 of our original 225 features, and for high values of 'outOf', ablation removed 216 features, resulting in 12 and 9 features, respectively:

```
final features for model fit to low 'outOf' values:
rating
price
outOf
userTotReviews
itemTotReviews
avgItemHelpfulness
userRatedReviews
category_Women
category_Petite
reviewTextNumSents
reviewTextFscore
reviewTextFKscore

final features for model fit to high 'outOf' values:
rating
unixReviewTime
price
outOf
itemTotReviews
avgUserHelpfulness
itemRatedReviews
category_Women
summaryTextNumWords
```

The final model achieved a Mean Absolute Error (MAE) of 0.1807 on the validation data, and an MAE of 0.17586 on the private test dataset.