Name: Laya Dang
BUID: U39822493

CS 506 Midterm: Amazon Movie Reviews Rating Prediction

Introduction

The goal of this project is to predict the star rating of Amazon Movie Reviews, given the number of helpful votes (HelpfulnessNumerator), total votes (HelpfulnessDenominator), Time, Text (user review), and Summary. There are 1,485,341 labeled data points to work with.

Feature Extraction

By far the most important aspect of this project is the feature extraction. After experimenting with a few different features, I chose to use the following:

- **Year**: only Year because Month looked uniformly distributed across the scores
- **Helpful, Unhelpful**: HelpfulNumerator, HelpfulnessDenominator – HelpfulNumerator to capture downvotes
- **SummarySentiment**: with lemmatized and cleaned summary text on NLTK's Vader
- **CleanedTextSentiment**: with lemmatized and cleaned full text on NLTK's Vader
- **ExclaimationCount**: number of exclamation marks
- **AllCapsCount**: number of words that were in all-caps
- **UniqueWords**: set of total words over total word count of full text
- **ProductAvgScore**: from training split, group ProductId and find average
- **UserAvgScore**: from training split, group UserId and find average
- **ProductPopularity**: from training split, total reviews product got
- **UserPopularity**: from training split, total reviews user made
- **TF-IDF** (`ngram_range = (1,3)`): ignoring terms that appear in over 55% of documents and in less than 1%, this is the TF-IDF matrix of both summary and full text

Exploratory Data Analysis

I start with a distribution of scores in the dataset:

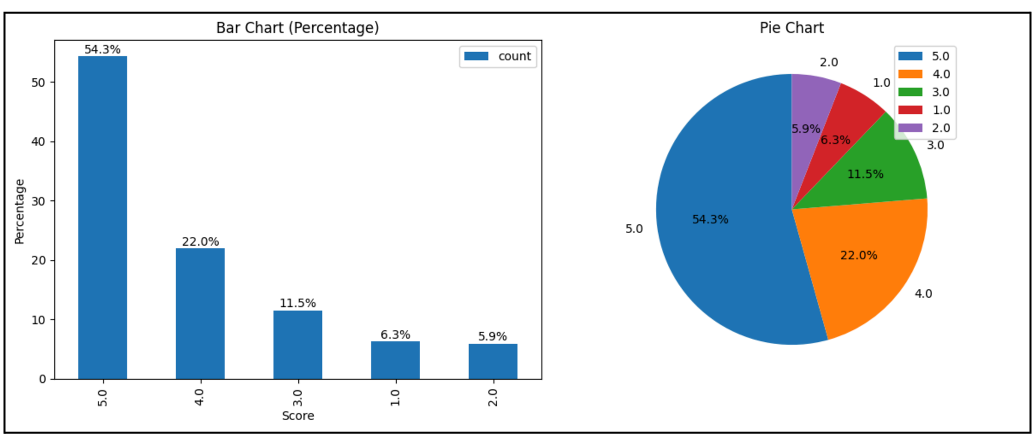

Figure 1. Distribution of Scores in Full Dataset

There is a clear skew towards higher scores. I decided to not balance the dataset on this feature, as this seems like a valid make-up for our submission dataset.

Name: Laya Dang
BUID: U39822493

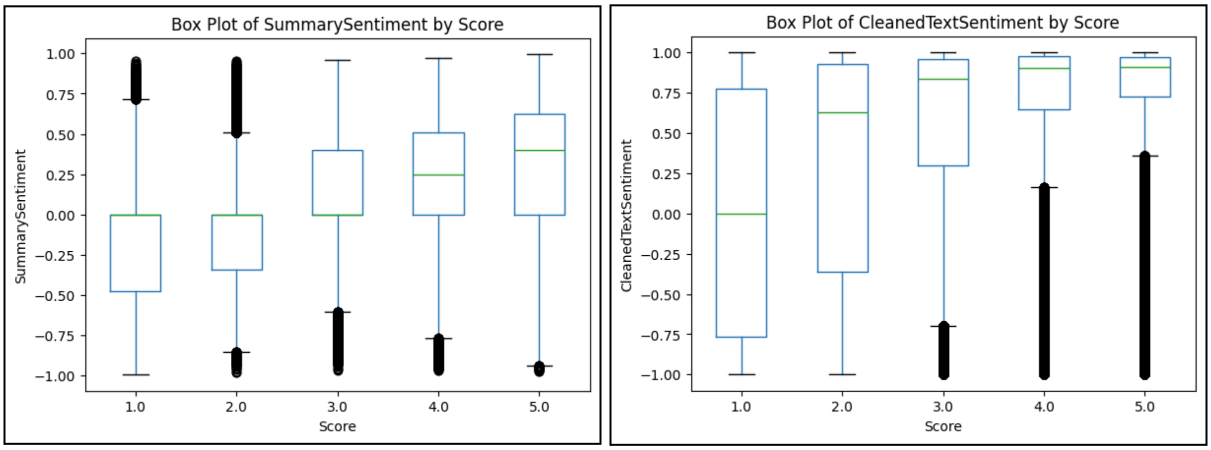

Figure 2. Box Plot of Summary and Full-Text Sentiment by Score

The most promising feature seems to be NLTK's Vader sentiment scores.[1] This library determines the overall sentiment and polarity of any text, trained on labeled text (mostly from social media), extensive lexicon dictionary, and grammatical rules.[2]

I initially did polarity (0 or 1 for positive and negative), but that did not capture enough of the text's nuance—at most differentiating between a 1- and a 5-star review. I decided to use the compound score instead, which as shown in Figure 2, shows much clearer differentiation between each rating.

Other features such as UserAvgScore, ProductAvgScore, Helpful, and Unhelpful also showed a similar distinction between each score category, with high variance, interquartile range, and outliers. See Figure 4 in the Appendix.

Model Selection

Since this is a multi-class classification problem with a large dataset and many features, my intuition was to try Logistic Regression, K-Nearest Neighbors, and Decision Tree models.

Due to time constraints, I resampled my dataset (undersample all classes and maintain the same ratio), tested the main parameters or the best parameter, and based on the best accuracy score, chose my model. These are the results:

- **Logistic Regression**: I tested C parameters (inverse of regularization strength) from 0.01 to 0.46. The mean cross-validation score was around 0.67, but accuracy was only 0.58 on the testing set, implying overfitting. It also took the longest to run out of all the models, so I gave up on this model pretty quickly.

---

[1] C.J. Hutto and Eric E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," *Eighth International Conference on Weblogs and Social Media* (ICWSM-14), Ann Arbor, MI, June 2014
[2] Ma, Ying. "NLP: How Does NLTK.Vader Calculate Sentiment?" Medium, February 5, 2020.

Name: Laya Dang
BUID: U39822493

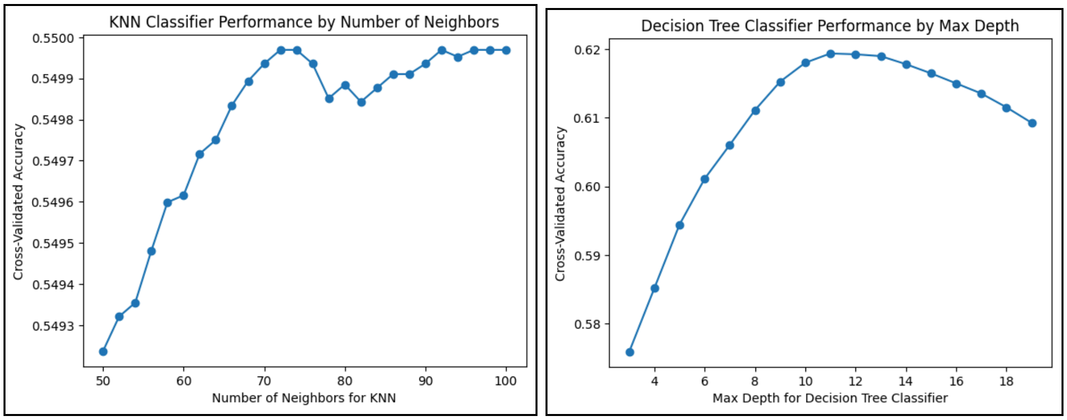

Figure 3. Testing Parameters (`n_neighbors` in KNN and `max_depth` in Decision Tree)

- **K-Nearest Neighbors**: I tested n_neighbors from 50 to 100 (step of 2) and found 72 neighbors worked best. The mean cross-validation score plateaus at 55% (left figure) with similar performance on the testing set.
- **Decision Tree**: I tested max_depth from 3 to 19, and found the best max depth was at 11 with 62% highest mean cross-validation. Performance was similar on the testing set.

Out of these models, I found decision trees worked best. After searching up ways I can improve this for the dataset, I found the XGBoost (Extreme Gradient Boosting) method, which helps models that include a mixture of weak features.

Applying XGBoost

Since we did not cover Gradient Boosting in lecture, I wanted to learn more about how this works. The algorithm starts with a base tree, built in the same way a typical decision tree is created. The difference is the "additive" nature of the algorithm, where subsequent new trees are created based on the previously added trees' mistakes. To score the tree, the residuals (difference between models actual vs predicted) are aggregated into a loss function.[3]

Since there are so many parameters and time constraints, I did a Random Grid Search for hyperparameter tuning, which tests a random combination of specified parameters and returns the best one. I ran this overnight since this took the longest.

Some further pruning I did was checking the most and least important features (measured as number of occurrence in the trees), drop the least important features, and transform the top features. See Figure 5 in Appendix for what the features are.

Conclusion

The best accuracy my model could reach was about 67%. The largest issue was classifying between mid-range reviews (2-, 3-, and 4-stars), so future work would look at what words distinguish these categories.

[3] IBM. "What Is XGBoost?" IBM, August 7, 2024. https://www.ibm.com/topics/xgboost.

Name: Laya Dang
BUID: U39822493

Reference

C.J. Hutto and Eric E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," Eighth International Conference on Weblogs and Social Media (ICWSM-14), Ann Arbor, MI, June 2014

IBM. “What Is XGBoost?” IBM, August 7, 2024. https://www.ibm.com/topics/xgboost.

Ma, Ying. “NLP: How Does Nltk.Vader Calculate Sentiment?” Medium, February 5, 2020. https://medium.com/@mystery0116/nlp-how-does-nltk-vader-calculate-sentiment-6c32d0f5046b.

Name: Laya Dang
BUID: U39822493

Appendix

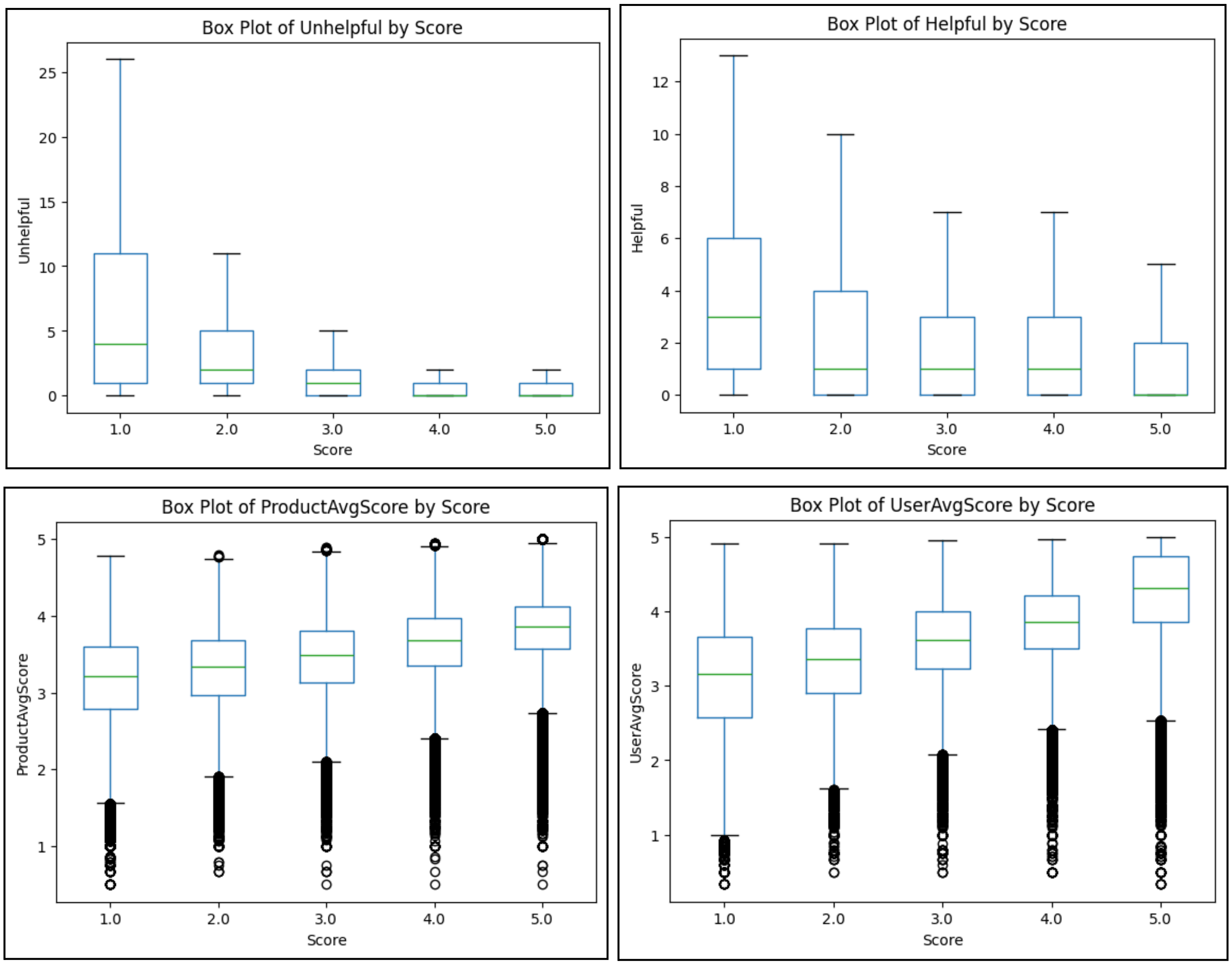

Figure 4. Box Plots of Various Features in the Training Set

Name: Laya Dang
BUID: U39822493

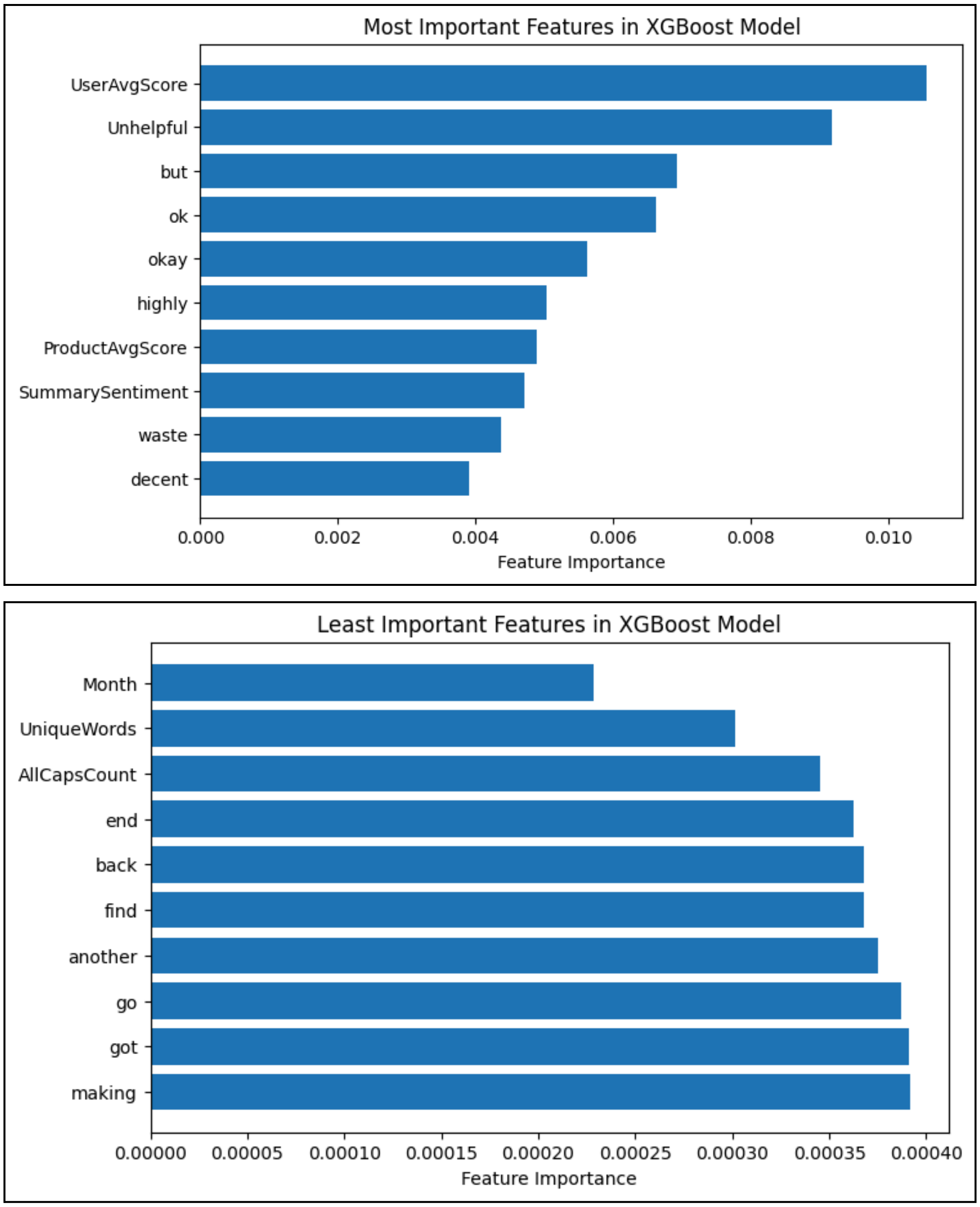

Figure 5. Most and Least Important Features according to XGBoost