Amazon Movie Review Rating Prediction

Overview

This project aims to predict star ratings of Amazon movie reviews based on various features extracted from the review data. Using a mix of text-based and interaction features, the model focuses on classifying ratings accurately with feature engineering, aggregation, and transformation techniques. This document outlines the final model's structure, key improvements, and observations noted throughout development.

Final Model and Feature Engineering

The implemented model ultimately only utilizes LightGBM as the primary classifier but there were many other classifiers that were used as well. The model incorporates engineered features based on the textual, categorical, and numeric information available in the dataset, described below.

Key Feature Engineering Steps

1. Text-Based Features

- Review Length, Summary Length and Word Count: Calculated the number of characters and words in both Summary and Text fields.
- Sentiment Analysis: Using the TextBlob library, sentiment polarity and subjectivity scores were derived for both Text and Summary. These features provide insight into the review's sentiment, capturing positive or negative language, and contribute to understanding the tone of the review.

2. Interaction and Aggregation Features

- Ratio Terms: The Helpfulness feature, calculated as the ratio of HelpfulnessNumerator to HelpfulnessDenominator, reflects the proportion of users who found a review helpful. This continuous value serves as a measure of community validation, indicating review quality and usefulness.
- Interaction Terms: Interaction terms such as LengthPolarityInteraction and HelpfulnessScoreInteraction combine review length with sentiment and helpfulness metrics. These terms reflect complex relationships that may impact the review score, such as a long, highly positive review often correlating with a high rating.
- User and Product Averages: For each UserId and ProductId, the average score was calculated to capture tendencies of specific users and overall product ratings. Missing values for new users or products were filled with the global average score.
- Log Transformations: ReviewLength and WordCount were log-transformed to address skewness, especially for longer reviews.

3. Temporal Features

- Year of Review: Extracted from the Unix timestamp, ReviewYear accounts for the potential shift in ratings over time. This feature helps capture temporal changes in user behavior or product popularity.

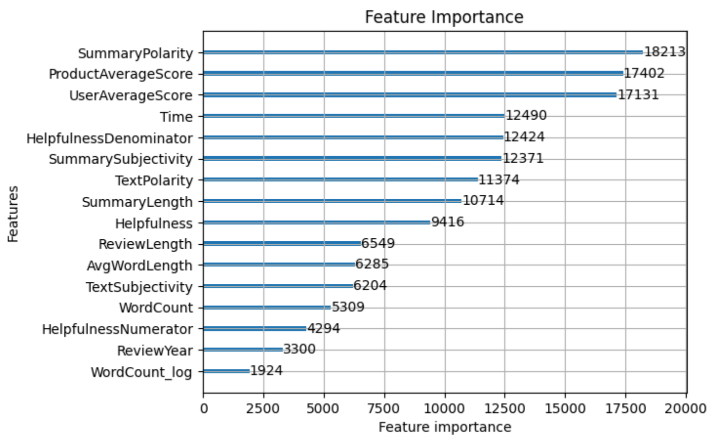
Feature Selection and Model Optimization

Feature selection was conducted using Lasso Regression to focus on the most predictive features and avoid overfitting. Additionally, Optuna was employed for hyperparameter tuning, and the following techniques were applied to improve model performance:

Special Tricks and Observations

1. LengthPolarityInteraction, HelpfulnessScoreInteraction and HelpfulnessScoreInteraction lead to overfitting: Through experimentation, my accuracy score was too high which shows that the model fitted too much to the training set. This makes sense since the Interaction features can have high complexity (the interaction features is obtained by multiplying two features) and high variability (small change in one feature leads to big change to the interaction feature).

2. Summary Polarity importance: This feature captures the overall sentiment of the review's summary, offering a concise, high-level indicator of the reviewer's opinion. This feature was the most relevant predictor as shown in the graph below.



3. User and Product Reputation Adjustment: Average scores per user (UserAverageScore) and product (ProductAverageScore) significantly improved model accuracy by capturing biases from reviewers and reputation effects on products. This insight was particularly valuable in cases where certain users consistently gave low or high ratings.

4. Temporal Adjustment for Review Year: Including ReviewYear provided a minor but consistent improvement. Older reviews tended to have higher ratings, possibly due to fewer total reviews or different rating standards over time.

5. ReviewMonth, ReviewWeek, ReviewDayofWeek was not included: Using Lasso, it showed that these features had low coefficients which means that they didn’t affect accuracy much. This makes sense since Review Year is enough to show the relevance.

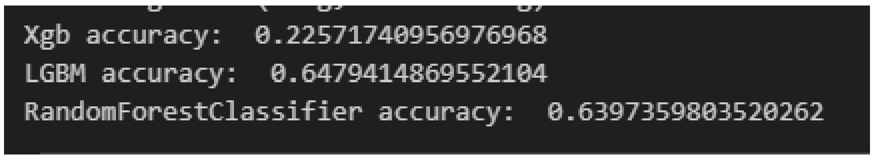
Modeling and Classifier Tuning

Though the lightGBM classifier was used at the end, I also experimented using a blend of XGBoost, RandomForest, and Logistic Regression classifiers which led to a stacking approach as well. I mainly focused on XGBoost and LightGBM as the primary classifiers due to their robustness with large datasets and feature interactions. GridSearchCV and Optuna were used for hyperparameter tuning, with specific attention given to:

- Learning rate and max depth in boosting models, balancing accuracy with the risk of overfitting.
- Feature Importance Analysis in boosting methods helped prioritize essential features like SummaryPolarity and UserAverageScore.

Note: the parameters were found using a smaller dataset of 10,000 points in the training set as it took too long finding the parameters with all of the training set data.

After evaluating every approach, I made multiple decisions on which classifier to use. It proved too time consuming to use the Stacking Classifier as it required multiple classifiers to be running. Also using XGBoost and Randomforest was too slow because they are not optimized for parallelization in the same way as LightGBM, it also gave less accuracy scores shown below. I ultimately decided to use the lightGBM classifier since it was faster than the other boosting methods and the parameters given from GridSearchCV allowed me to prevent overfitting.



```
Xgb accuracy:  0.22571740956976968
LGBM accuracy:  0.6479414869552104
RandomForestClassifier accuracy:  0.6397359803520262
```

Conclusion

Through feature engineering, selection, and model tuning, the final algorithm effectively classifies review ratings with improved accuracy. The combined use of sentiment analysis, interaction features, and reputation-based adjustments led to a model capable of leveraging subtle patterns in the data. This approach illustrates the value of exploring deeper relationships within features, particularly when dealing with nuanced data like textual reviews. Future work may explore incorporating NLP techniques such as word embeddings for enhanced text feature representation.