CS506 Midterm Report

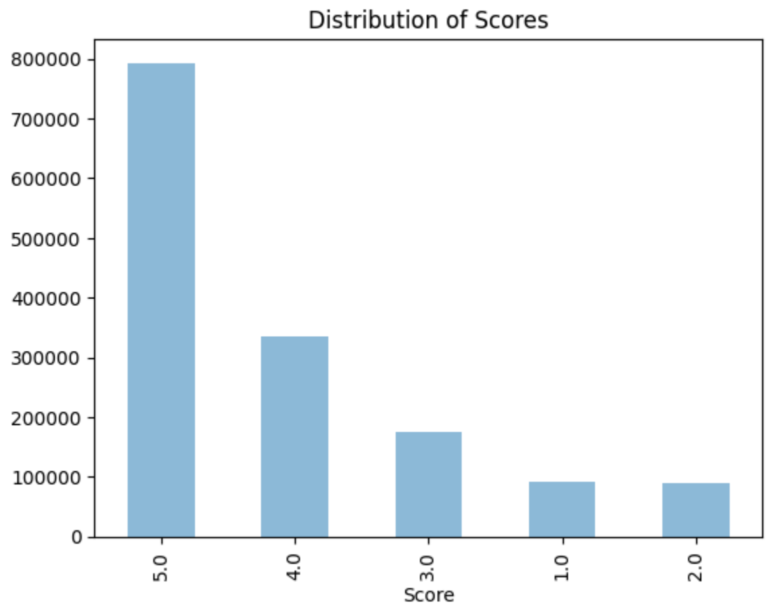
Isa Alsafwah (isaa@bu.edu)

Introduction

In this assignment, we are given two main csv files, train.csv which contains 1.71GB worth of data and test.csv. Both of which involve data dealing with users on Amazon Movie Reviews.

Preprocessing

Score Distribution: A bar plot of the score distribution revealed an imbalance among the scores, with some scores more frequent than others. This observation guided further adjustments in feature engineering and model selection to better accommodate class imbalances.



Feature Engineering

Sentiment Analysis

To capture sentiment nuances in reviews, I used VADER, a sentiment analyzer for text. For both the review Text and Summary fields, VADER's compound sentiment score was calculated, which resulted in two essential features:

- Sentiment Score: Captures overall sentiment intensity of the review text.
- Sentiment Summary: Measures the sentiment in the review summary.

These features were essential as they quantified the positive, neutral, or negative tone of reviews, providing valuable predictive power for score prediction.

User-Behavior Related Features

Upon analyzing user behavior across multiple reviews, a few patterns in scoring consistency and reliability were realized. Using these insights, I created a few features, the key ones being:

- **User Avg Score**: Average score given by each user, capturing their general rating tendency.
- **User Standard Deviation (Std) Score**: Standard deviation of scores given by each user, indicating how consistently they rate.
- **User Review Count**: Number of reviews provided by each user, reflecting their level of activity.

Using these new features, it allowed for an improvement in prediction accuracy by accounting Max Iterations:for individual scoring behaviors, For example, those who reviewed products frequently exhibited more consistent rating patterns, influencing the model’s reliability.

Review-Specific Features

I pulled a few additional features from the review content itself to capture aspects like helpfulness, length, and sentiment dynamics:

- **Helpfulness**: Ratio of HelpfulnessNumerator to HelpfulnessDenominator, indicating how useful a review was perceived by others.
- **Review Length**: Number of characters in the review text, reflecting the depth of information provided.
- **Word Count**: Total words in the review text, useful for distinguishing short opinions from detailed feedback.
- **Summary Word Count**: Word count of the review summary.

<u>Feature Selection</u>

I applied SelectFromModel with a **RandomForestClassifier** as a pre-selector to identify features that significantly contributed to the target variable. By using a tree-based estimator, I could rank features based on importance and eliminate less relevant ones, effectively reducing dimensionality. The final selected features included user-based metrics, sentiment scores, and interaction features, which provided a compact and powerful feature set.

Model Selection and Hyperparameter Tuning

Then, I used HistGradientBoostingClassifier for its efficiency with high-dimensional data. Gradient boosting models are effective in capturing non-linear relationships in complex datasets, making them well-suited for text and user behavior data. The following key hyperparameters were optimized:

- Set to 100 to balance model complexity with computational efficiency.
- Learning Rate: Adjusted to 0.1 to prevent overfitting and ensure smooth gradient descent.
- L2 Regularization: Used to reduce overfitting by penalizing larger coefficients.

Works Cited

https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://en.wikipedia.org/wiki/Random_forest

https://en.wikipedia.org/wiki/Sentiment_analysis

https://www.geeksforgeeks.org/python-sentiment-analysis-using-vader/