Introduction

In recent years, the growth of e-commerce has produced an immense amount of user-generated content in the form of product reviews. These reviews offer valuable insights, but analyzing such large datasets effectively requires robust analytical methods. This project aims to predict star ratings for Amazon Movie Reviews using metadata and text content associated with each review. The dataset contains 1,697,533 unique reviews, and we aim to develop a model to predict the star ratings accurately. Various features such as product ID, user ID, helpfulness scores, and review text are used in the modeling process. The main goal is to leverage feature extraction and machine learning techniques to classify reviews effectively into their respective rating categories (from 1 to 5 stars).

Preliminary Analysis / Exploration

To understand the data better, preliminary visualizations and analysis were conducted. Key features such as "Score," "HelpfulnessNumerator," and "Text" were analyzed to gain insights into review patterns. Notably, a significant bias was found in the "Score" column, with a large percentage of reviews being rated as 5 stars. This imbalance suggests potential challenges for model training and classification accuracy.

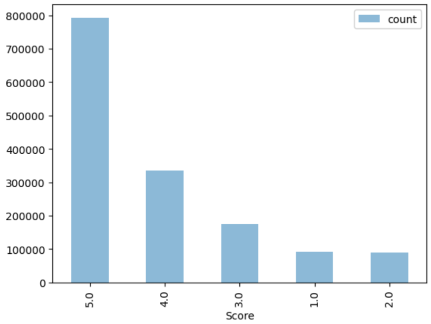

Figure 1: Score Distribution Count

Additional analysis included examining the relationship between helpfulness scores and review ratings. Various visualization techniques, such as bar plots and heatmaps, were employed to illustrate the distribution of ratings and correlations between features. The visualizations (Figures 1, 2, and 3) were not included in the final submission or notebook as they were generated during the exploratory step, and maintaining a concise notebook was crucial due to computational limitations, including RAM usage and runtime efficiency.

Another insightful discovery was examining the relationship between helpfulness scores and review ratings. Figure 2 illustrates that reviews rated as 1 star often have higher helpfulness ratings compared to those rated as 5 stars, suggesting a stronger emotional response from users for lower-rated products. Various visualization techniques, such as bar plots and heatmaps, were employed to illustrate the distribution of ratings and correlations between features.

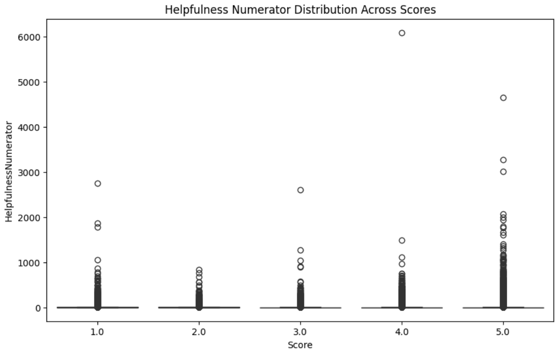

Figure 2: Helpfulness Numerator Distribution Across Scores

A pairplot visualization of a sample of the dataset was also conducted to explore relationships between numerical features such as "HelpfulnessNumerator," "HelpfulnessDenominator," "Time," "Helpfulness Ratio," and "Text Length." The pairplot indicated that most features had weak correlations, with a few exceptions, such as the correlation between "HelpfulnessNumerator" and "HelpfulnessDenominator." These insights informed feature engineering and selection for modeling (see next page for figure 3).

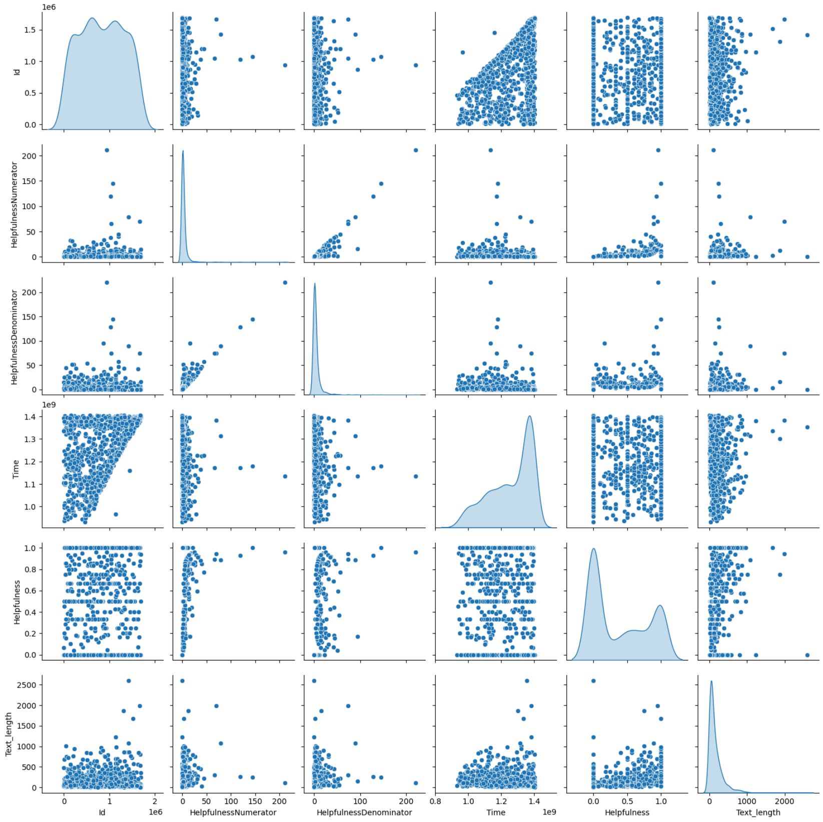

Figure 3: Pairplot of Numerical Features

Prediction / Modeling

Logistic Regression was chosen for this prediction task due to its interpretability, computational efficiency, and its ability to handle a large number of features effectively. Given the high dimensionality of the dataset, Logistic Regression provided a robust solution that allowed efficient training without extensive computational requirements. Additionally, it offers a probabilistic framework, which makes it easier to understand the influence of different features on predictions. This transparency is valuable when analyzing how factors like review text and helpfulness scores contribute to the final rating prediction. Considering the constraints on computational resources and time, Logistic Regression was also optimal due to its faster convergence compared to more complex models like Random Forests or Gradient Boosting, which can be more computationally intensive.

The primary evaluation metric was accuracy, with a score of around 0.557 achieved for the test set. Although this score may seem moderate, several limitations, such as class imbalance and the large variability in review content, contributed to the challenges of prediction accuracy.

Feature extraction played a crucial role in improving model performance. Text features were processed using TF-IDF vectorization, and dimensionality reduction was applied via Truncated SVD to limit computational resources. Categorical features, such as product and user IDs, were encoded using One-Hot Encoding, and numerical features were scaled for better normalization. These techniques were critical for preparing the data effectively for the machine learning model.

Methods ("Special Tricks") and Assumptions

To enhance model performance, several unique methods and optimizations were applied during the feature extraction and modeling stages:

1. Dimensionality Reduction with Truncated SVD: TF-IDF vectorized features for text data were reduced to 100 components using Truncated SVD. This helped to control computational complexity and avoid overfitting, allowing the model to focus on the most informative features.

2. Handling Class Imbalance: The dataset had a significant imbalance, with a large proportion of 5-star reviews. To mitigate this, the majority class (5-star reviews) was downsampled by 50%. This balancing technique helped prevent the model from being biased towards the majority class.

3. Regularization for Logistic Regression: The regularization strength parameter (C) was adjusted to 0.1, and `class_weight='balanced'` was used to give more weight to underrepresented classes. This approach aimed to improve the model's generalization capabilities and ensure that all classes were treated fairly during training.

4. Feature Engineering: A helpfulness ratio feature was added by dividing the "HelpfulnessNumerator" by the "HelpfulnessDenominator." This feature aimed to quantify how helpful a review was, irrespective of the number of people who voted, which provided additional context to the model.

5. Sentiment Analysis: Text features were enriched by applying sentiment analysis using polarity scores. The SentimentIntensityAnalyzer from the `nltk.sentiment.vader` package was used to assign sentiment scores, which contributed as an additional feature in distinguishing between positive and negative reviews.

6. Limiting Iterations for Efficiency: The maximum number of iterations for Logistic Regression was reduced to 500 to ensure that the model ran efficiently within a limited timeframe, while still converging to a good solution.

Conclusion

The process of predicting Amazon movie review scores involved multiple steps of data exploration, feature engineering, and model optimization. Despite the challenges of class imbalance and large variability in the review content, the use of Logistic Regression along with thoughtful feature extraction and optimization resulted in a model with an accuracy score of approximately 0.557. While there is room for further improvement, the methods employed—including sentiment analysis, regularization, and class balancing—contributed to building a robust baseline model that effectively captures important patterns within the data.