1. **Introduction**

In this project, the task was to develop a model to predict star ratings associated with user reviews from Amazon Movie Reviews, using features provided in a CSV file. The primary objective was to successfully process and analyze the dataset through data preparation, feature engineering, model selection, and evaluation, ultimately achieving a prediction accuracy above 58%. One major challenge was the large size of the dataset, which led to system crashes during processing, especially when converting text data into numerical features. To address this, I applied random sampling on the CSV file for iterative development and testing, reducing memory usage and improving efficiency in the testing cycle. The solution centers on two algorithms: Random Forest Classifier (the one I chose for my final submission) and XGBoost. I also employed special optimizations in feature engineering, text preprocessing, and dimensionality reduction to improve model performance while handling computational limitations.

2. Algorithm Development

Initially, I experimented with a simple text preprocessing method, TF-IDF (Term Frequency-Inverse Document Frequency), to convert the textual content of reviews into numerical representations. TF-IDF measures the importance of a word in a document relative to a collection of documents by decreasing the weight of commonly used words and increasing the weight of words that are less common. For a baseline, I began with a Random Forest Classifier with no parameters, due to its effectiveness in handling high-dimensional data and predicting categorical variables. During data exploration, I observed high variability in the text lengths and HelpfulnessRatio of reviews, missing data values, and class imbalances among the different

ratings. Processing the entire dataset with TF-IDF led to system memory crashes due to the high-dimensional output. This impacted my approach significantly, leading me to integrate dimensionality reduction techniques, such as Truncated SVD, and down-sampling of the data.

3. Special Optimizations and Techniques

To enhance interpretability and performance, I engineered the following features:

- FullText: A combination of Summary and Text columns, capturing the complete text for each review. This new feature allowed more expressive text analysis as I didn't want to lose any valuable information from Summary but also didn't want to have to process it on its own. I discarded the original columns to reduce memory usage.
- HelpfulnessRatio: Calculated as HelpfulnessNumerator / HelpfulnessDenominator, providing an indicator of how helpful a review was.

Text data was preprocessed using the NLTK library to remove stop words and punctuation, which helped reduce noise. The FullText column was vectorized using TF-IDF with limited features (max_features=500 and max_df=0.05). To address the high-dimensional output from TF-IDF, I applied Truncated SVD to reduce the feature matrix to 50 components. This dimensionality reduction minimized memory load while preserving the most impactful features. For Random Forest, I optimized the model by tuning hyperparameters. Through GridSearchCV I found the best parameters to be: max_depth=12, min_samples_leaf=4, and n_estimators=300. This helped balance the depth of trees with generalization, improving cross-validation scores. I used cross-validation on subsets of data to ensure stable performance. With XGBoost, I adjusted the eval_metric to mlogloss to suit multi-class classification. To handle sparse features from the TF-IDF matrix and mitigate potential class imbalance, I used class weighting in the Random

Forest. This approach helped the models remain sensitive to the minority classes, which were those of 3 stars or below and improved the overall predictive accuracy.

4. Assumptions and Their Impact

Key assumptions included:

- HelpfulnessRatio Correlation: Higher helpfulness would correlate positively with the star rating, an assumption that justified including HelpfulnessRatio as a primary feature.
- TF-IDF as Feature Importance: The TF-IDF vector would provide sufficient insights into review sentiment and context, potentially predicting user sentiment effectively even without complex sentiment scoring techniques.
- Dimensionality Reduction Sufficiency: The SVD-reduced TF-IDF matrix retained meaningful information without compromising prediction accuracy.

These assumptions guided the design, especially in feature selection and processing steps. By treating HelpfulnessRatio and the reduced TF-IDF matrix as core features, I could manage computational costs effectively while maintaining model relevance to the task.

6. Conclusion

Patterns such as frequent words correlating to specific star ratings reinforced the need for textual based features. Given that the system frequently crashed during TF-IDF processing, I addressed this by down-sampling data for iterative testing and incorporating early-stage dimensionality reduction. I experimented with different n-gram ranges and feature limits to identify optimal TF-IDF settings for computational feasibility. Integrating Truncated SVD with models like Random Forest and XGBoost yielded significant accuracy improvements. The Random Forest model that I tuned in the end performed much better than the XGBoost based on their classification reports and cross-validation scores.