1. Main Goal

To be honest, I have never worked on such a large dataset and I am afraid it would take more than 3 hours to run a single model. As a result, the primary goal is to keep both efficiency and accuracy. I will explain in further detail how I made it.

2. Data Processing and Feature Engineering

The first step in the project was to explore the dataset and understand its patterns. During this exploration, it was evident that the dataset was heavily imbalanced, with a significant number of reviews rated as **5 stars** compared to other ratings. This imbalance was something we needed to consider in subsequent steps.

Besides, to reduce memory usage, the data types of columns were specified at the time of loading.

Since text was expected to be an important feature, I created a new feature called **SentimentScore** using sentiment analysis. Initially, I consulted ChatGPT, which suggested the `Afinn` library. Sentiment analysis was conducted using the `Afinn` library to calculate a sentiment score for each review text. A batch approach was implemented to efficiently score reviews in chunks, given the large dataset size.

After creating these features, I constructed the **X_train** for model training and **X_submission** for final submission.

3. Feature Selection

To avoid introducing noise or irrelevant features, a feature importance analysis was conducted. Using a preliminary RandomForest model, features such as **Helpfulness Ratio**, **Text Length**, **Sentiment Score**, and **Product/User Average Scores** were found to be significant predictors.

Features like **"Year"** and **"Summary Length"** were tested but eventually removed due to their negligible contribution to predictive power.

4. Model Selection and Training

The initial model selection involved trying multiple models to see which would perform best given the large dataset size and the need for both efficiency and accuracy. I first tried **Naive Bayes**, but it proved inadequate for such a large dataset.

The final model chosen for this task was **RandomForestClassifier**, owing to its robustness against noisy features and ability to handle a large number of data points effectively. Several reasons supported this choice: Handling Missing and DataInterpretable Feature Importance.

Given the large parameter space, **GridSearchCV** was employed for hyperparameter tuning. The key parameters tuned included:**n_estimators, max_depth and class_weight**. To reduce computation time during hyperparameter tuning, I created a sample dataset consisting of only 10 % of the training data. Once optimal parameters were identified, the entire dataset was used to fit the final model.

In order to handle the majority of reviews rated at **5 stars**. I employed **Class Weights Adjustment** in the RandomForestClassifier (`class_weight='balanced'`) ensured that misclassifying minority classes was penalized more than majority classes during training.

6. Evaluation and Insights

The model achieved an **overall weighted F1-score of 0.68**, with strong performance on the majority class (5-star ratings) and moderate improvements on under-represented classes (1, 2, and 3-star ratings). The following were noted during evaluation:

```
Classification Report:
              precision    recall  f1-score   support

         1.0       0.54      0.55      0.54      1908
         2.0       0.26      0.15      0.19      1723
         3.0       0.31      0.22      0.26      3525
         4.0       0.37      0.29      0.33      6711
         5.0       0.71      0.86      0.78     15840

    accuracy                           0.59     29707
   macro avg       0.44      0.41      0.42     29707
weighted avg       0.55      0.59      0.57     29707

Confusion Matrix:
 [[ 1040   246   257   169   196]
 [  397   261   434   323   308]
 [  267   257   792  1070  1139]
 [  143   164   642  1949  3813]
 [   96    89   413  1691 13551]]
```

- **High Precision for 5 Stars**: The model achieved good precision and recall for 5-star ratings, indicating that the majority class was correctly captured.
- **Improvement for Lower Ratings**: Precision and recall for 1 and 2-star reviews remained challenging due to their low representation.
- **Sentiment as a Key Indicator**: It was assumed that the sentiment score of a review text would be highly correlated with its rating. This assumption held true in many cases, but noise in the sentiment analysis for neutral reviews contributed to some misclassifications.
- **Helpfulness as a Trust Metric**: Reviews with higher helpfulness were assumed to be more reliable and predictive of rating quality. However, it was found that helpfulness was not always correlated with extreme ratings, and thus its predictive value was moderate.

7. Conclusions and Future Work

The RandomForest model, enhanced through thoughtful feature engineering, hyperparameter tuning, and class imbalance handling, provided a reasonably good solution for the Amazon Movie Reviews rating prediction.

The main model I have chosen is random forest with testing method https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html

https://scikit-learn.org/1.5/modules/model_evaluation.html

https://scikit-learn.org/1.5/modules/grid_search.html

http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=6010