Write-Up

I don’t have extensive knowledge about NLP, so I essentially approached this competition as a general classification challenge.

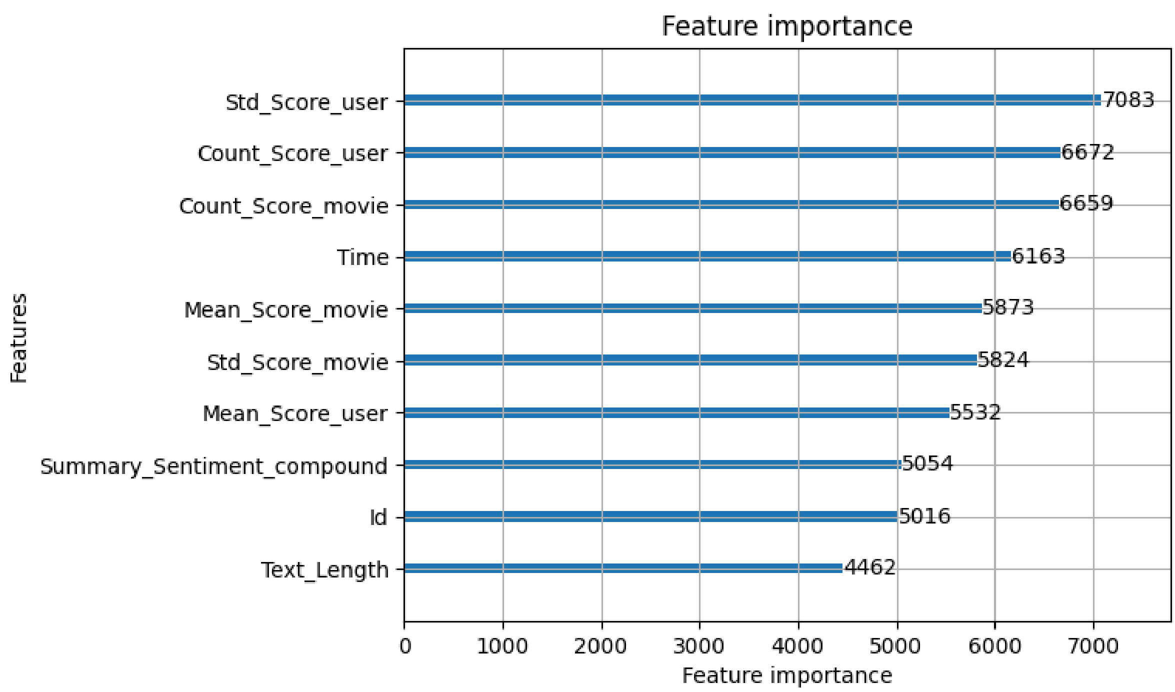
Helpfulness

The first feature I added was "Helpfulness" since it’s easy to implement and free!

User Preference and Movie Averages

One of the most important tasks I undertook was grouping the ratings by `userID` and `movieID`. This allowed me to derive several features:

- **Movie Features:**
  - **Average Rating:** This reflects the general quality of the movie.
  - **Count of Ratings:** This indicates the popularity of the movie.
  - **Standard Deviation (STD):** This measures the variability in ratings, helping the model to discern whether to weight the semantic analysis of individual reviews more heavily or to rely on the average rating.
- **User Features:**
  - **Average Rating:** This reflects the user’s personal rating tendencies.
  - **Count of Ratings:** This provides insight into the user’s experience and qualifications.
  - **Standard Deviation (STD):** This indicates the range of ratings a user typically gives. For instance, some users, like myself, tend to score movies between 2 and 4, categorizing films as either masterpieces or total failures.



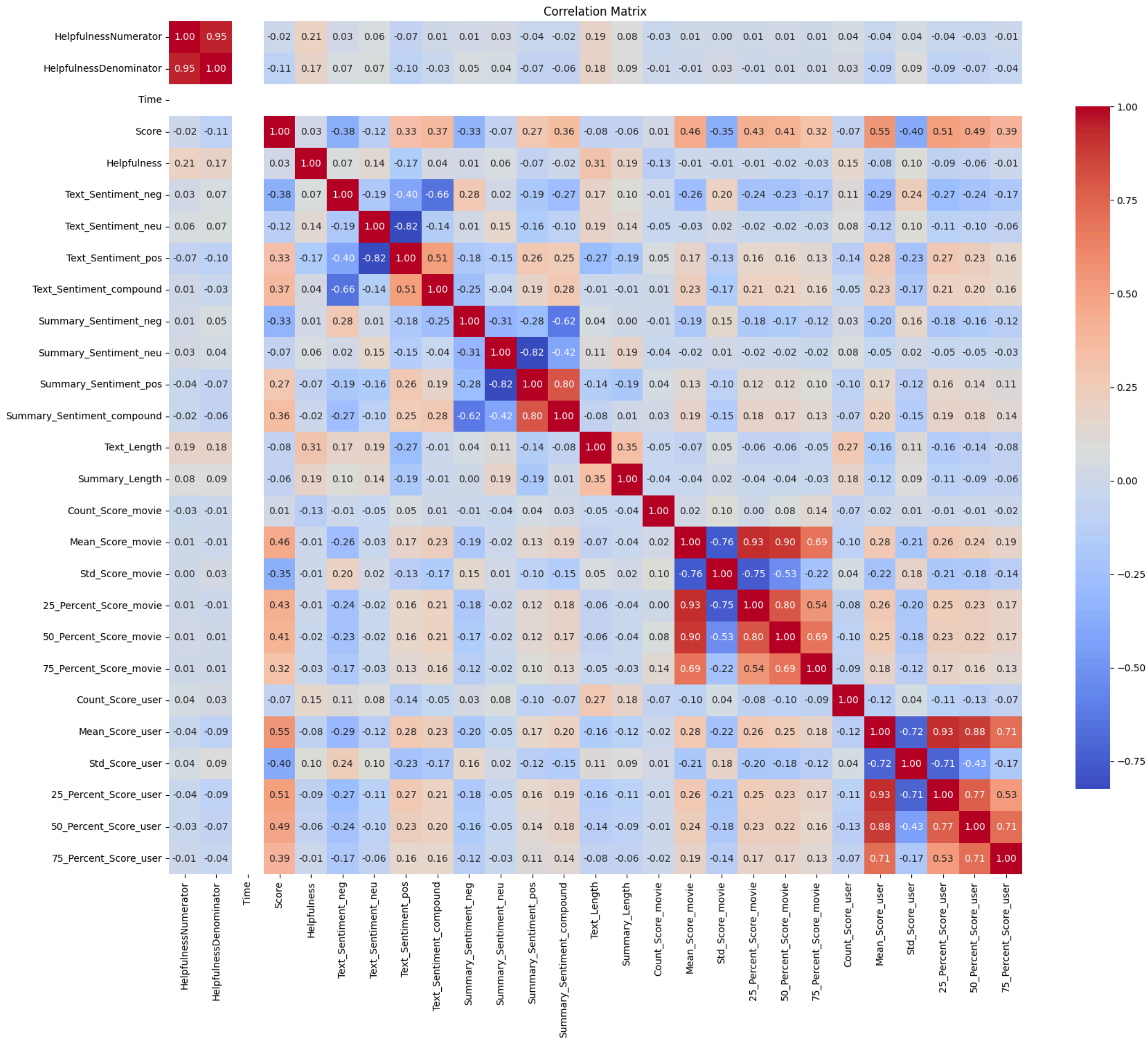
Label Encoding

I label-encoded `ProductId` and `UserId` since they were originally strings. I hoped this would aid in classification, although ultimately, it didn't significantly impact the results.

Time as a Feature

By analyzing feature importance plots and correlation matrices, I found that `Time` emerged as an important feature. My current hypothesis is that it may reflect the economic conditions when the movie was released, which could correlate with the quality of the movies.

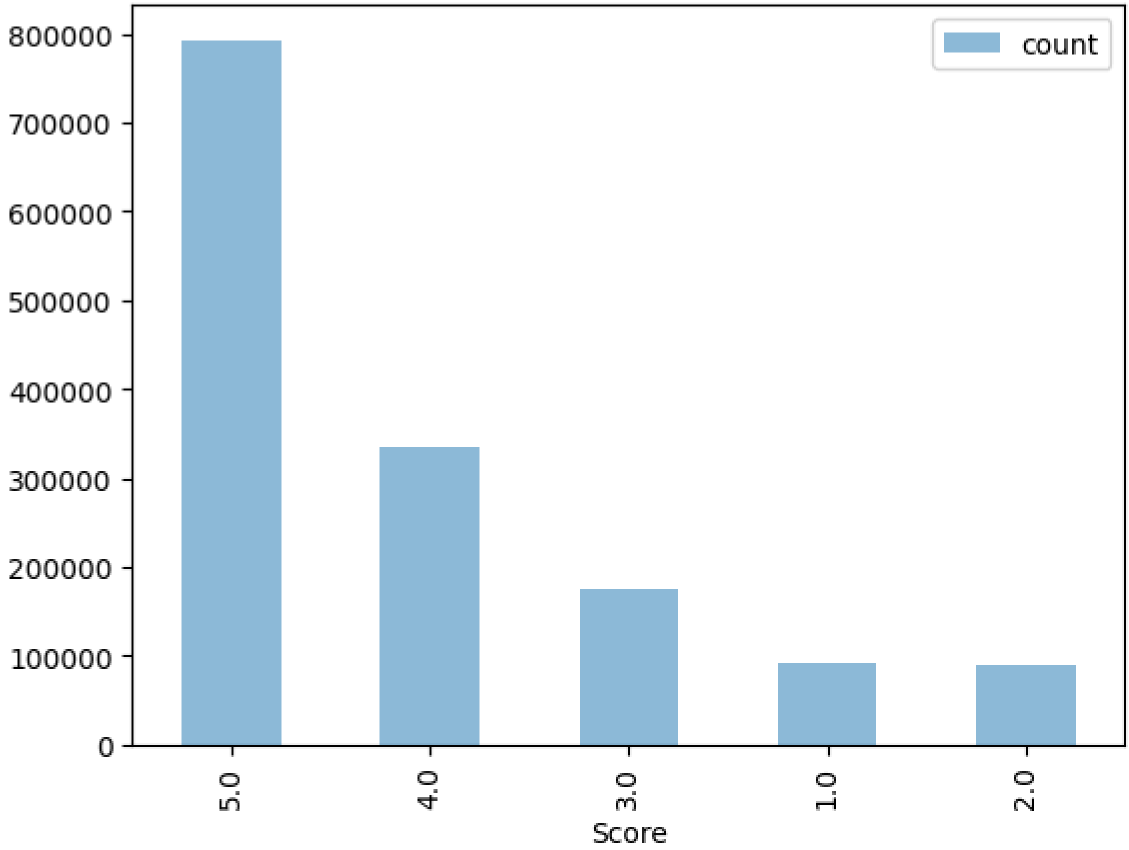
- Here's the Correlation Matrix and you can see that I had already dropped some of the features. (Q1, Median, Q3)



NLP Features

The only NLP features I included were the sentiment scores of the `Summary` and `Text`, as well as their lengths. I believe that longer reviews are often more thoughtful and serious.

Addressing Class Imbalance



Upon observing the distribution plot, I discovered that the dataset was extremely imbalanced. To address this, I employed SMOTE to oversample the features mentioned earlier in the 1 to 4 star comments, aligning their numbers with those of the 5-star ratings. Fortunately, this strategy improved model performance.

Classification vs. Regression

I typically prefer regression techniques, clamping scores into integers since discrete ratings often maintain a meaningful relationship. However, for this competition, I opted for classification. My previous models tended to categorize 4-star ratings as 5 stars, which misrepresented their distinctiveness. By treating them as separate categories, the model learned that predicting an actual 5-star rating as 4 stars wouldn't yield any credit, as there's no concept of partial correctness in this context.

Models Used

I experimented with XGBoost, LightGBM, and Extra Trees, applying aggressive hyperparameter tuning throughout the competition. The best results came from XGBoost. While I generally prefer LightGBM due to its advantages, it is also prone to overfitting. Given the time constraints, I focused more on feature engineering rather than model selection. The benefit of XGBoost lies in its layer-by-layer development, which can help mitigate overfitting compared to other methods, and that's important because I had already overfitting my model with only 10% of the dataset or many time.

Hyperparameter Tuning

I utilized GridSearchCV for tuning, exploring the relationships between key parameters like `n-estimators`, `max-depth`, and `learning_rate`, which are crucial for optimizing XGBoost.

Attempt to Combine Multiple Models

I attempted to use another vote classifier (I don't want to train 3 heavy models in a row), but it resulted in the worst performance. The combined predictions were even less accurate than those made by the individual models, LGBM and XGBoost. I had hoped that these models would balance out each other's biases, but it seems they ended up making the same mistakes together!