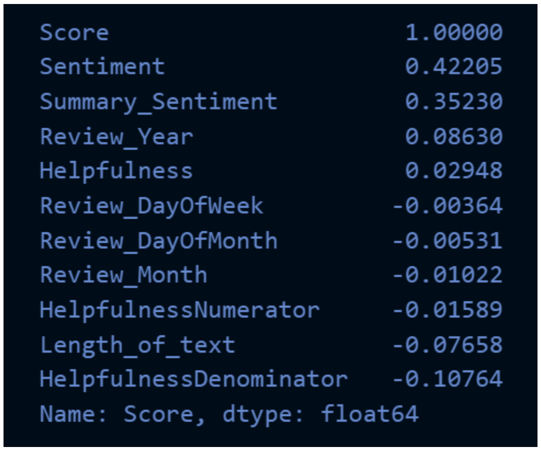
Amazon Movie Review Writeup

Task:
Predicting customer review ratings based on textual and numerical data.

Approach:
In the preliminary analysis of the data, I tried to observe how features in the dataset could influence the review score. I first thought that the time field should be converted to date format in order to observe, including but not limited to review year and review month. Oftentimes the written reviews for a product is a good indicator of what a customer might think about a product so the text fields were another factor that I thought could be helpful in our prediction model. The challenge then lies in how I can use features and engineer them in a way that captures user/rating tendencies, after which I will use my findings to fit multiple models and determine which one works best.

Data Processing/Feature Selection:
In preprocessing the data, I filled in null values and ensured that my features were formatted correctly. Then I ran a correlation test for each feature against the score field to see whether the relationship was strong enough that it would contribute positively to the model, rather than introducing noise. From below, I thought that the time feature would be unhelpful as the time translated features had little to no correlation. And clearly the engineered text feature, sentiment has a strong correlation, thus it should be used to train our model. For my model I used vader from the NLTK module to extract this sentiment score

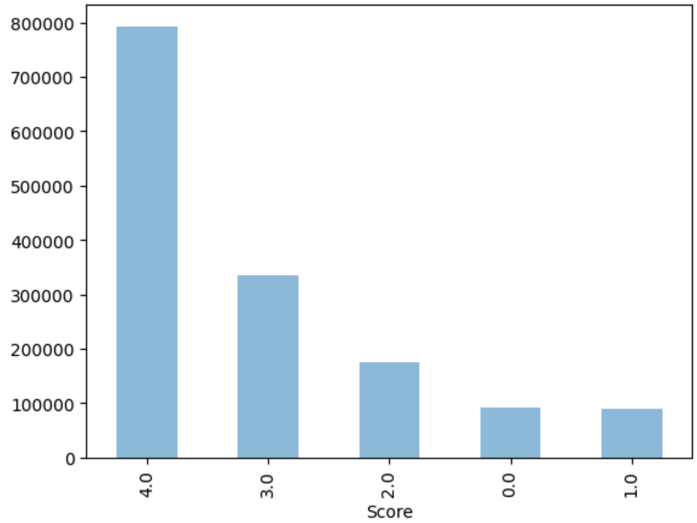


```
Score                     1.00000
Sentiment                 0.42205
Summary_Sentiment         0.35230
Review_Year               0.08630
Helpfulness               0.02948
Review_DayOfWeek         -0.00364
Review_DayOfMonth        -0.00531
Review_Month             -0.01022
HelpfulnessNumerator     -0.01589
Length_of_text           -0.07658
HelpfulnessDenominator   -0.10764
Name: Score, dtype: float64
```

(In a different version of my model)

Later, I realized that each user could have a pattern in how they score products, and decided to capture their mean score, typical score deviation and how many reviews they have given. This is useful because it could help account for bias in ratting each user, allowing us to make a more reasonable guess on what a given user would predict on any general product.

Some other features I included were the helpfulness features and length of text features derived of each review to capture the nuances of how relevant a text review may be to the review itself.



Our data is skewed heavily to the five star reviews. So it would be important to take that into consideration for capturing the details that separate a lower rated movie to the 5 star ones, as the model is more likely to predict a 5 star review than anything else. Each index is (review star -1).

Model Selection:

KNN Classifier - This was the first model I tried fit my data with as it was what was provided to us. I tried to use different K-values to see how far that would improve my model. My best attempt was about 53% accuracy locally. At this point I had set my K-value to 20 and also was incorporating Time features.

RandomForest Classifier - In this second model, I was changing the criterions and n-estimators to see what were the optimal parameters. At n=300 and criterion = entropy, I achieved this model's best result of about 55% accuracy locally and 55% on kaggle.

XGBClassifier - This is the last model I tried and yielded the best result of about 64% locally, and about 60% on kaggle. I believe this approach was most successful because it is better out of the three in handling non-linear relationships.