Predicting Amazon Movie Review Ratings

Data Description

The dataset provided to us consisted of Amazon movie reviews each of which corresponded to a rating of 1-5. The dataset consisted of columns, Id (unique review identifier), Product Id (unique product identifier), User ID (unique user identifier), Time (timestamp of review), HelpfulnessNumerator (number of people who found a review helpful), HelpfulnessDenominater (number of people who found a review helpful or unhelpful), Summary (summary of review), Text (actual review) and Score (rating between 1-5).

Data Cleaning

In the data cleaning process, I performed the following:

1. Set data type of Helpfulness columns to float
2. Dropped rows where HelpfulnessDenominator was 0 or NaN
3. Set data type of Text and Summary columns to str
4. Filled entries with NaN in place of Text and Summary with ‘ ’.

Sampling

Given the enormous amount of entries in the data, sampling it was pertinent to ensure compute and memory resources were not overloaded. The sampling was done after the test train split to preserve the original distribution of data in the test set. Another reason for sampling was to balance out different score classes in training data.

Initial approaches:

1. Initially I took a stratified sample which had the same distribution as the original data. However after performing experiments I found out that 90% of the predictions of the model were equal to 5.
2. My second approach was to take a completely balanced sample but that resulted in an even poorer accuracy than 1.

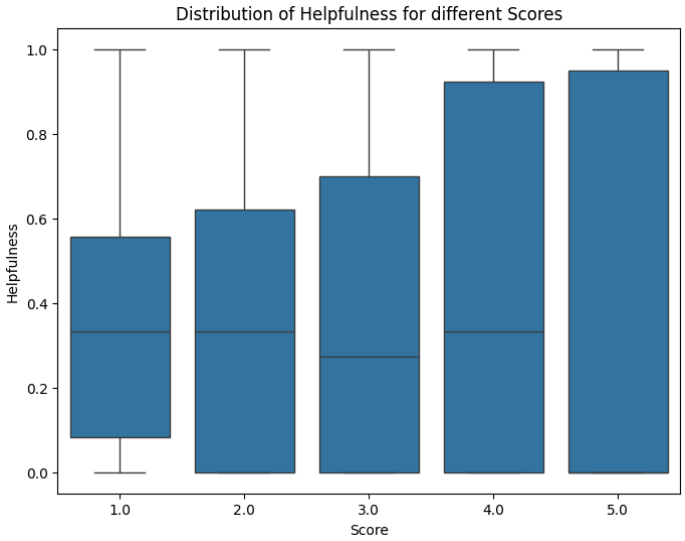
Final approach: I manually tuned the amount of entries taken for each score in the training set. I started with a total of ~ 15000 entries with 5 being the majority to reflect the pattern in the original set. However other classes were given more visibility than in the original data. Final sample amounts were { 5 : 15000, 4.0 : 10000, 3 : 8000, 2 : 6000, 1 : 4000}. Increasing the entries further did not yield a better result.

Data Preprocessing

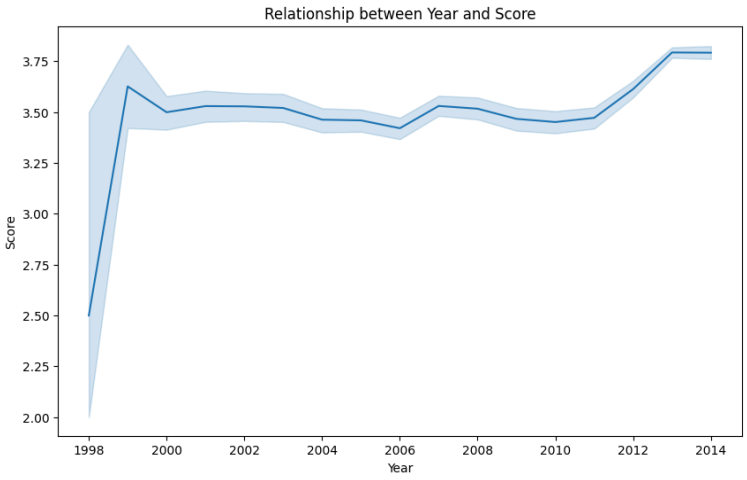
In this phase, I preprocessed the Text and Summary columns by removing punctuations, removing stop words, converting to lowercase, and performing lemmatization.

Data Exploration

From my exploratory analysis I discovered that helpfulness did not have strong correlation with score (corr coefficient = 0.049). Although there are some pattern changes in helpfulness as score increases, I confirmed through my experiments that helpfulness did not positively affect the accuracy.



I also explored the relationship between Time and Score which was again, not significant. Corr coefficient = 0.08



Feature Engineering

I attempted multiple things in this phase, some of which were successful.

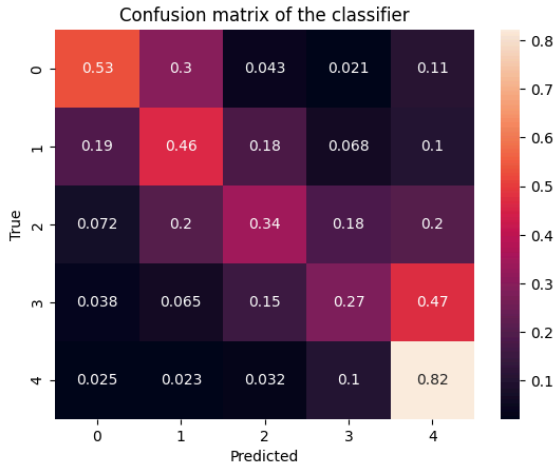
Successful: I performed count vectorization on both the Text and Summary columns to capture the frequency of words and phrases. I utilized at first unigrams, then bigrams, then trigram. Increasing the number of grams led to an increase in accuracy from 55 % to 60 %. The count vectors were input to my model.

Unsuccessful:

1. I created a Helpfulness column which did not help with accuracy at all.
2. I also performed TFIDF vectorization on Text and Summary columns with max features equal to 5000. I then input it to my model which only yielded an accuracy of ~50%. I tried to reduce dimensionality by first performing PCA but that repeatedly crashed my system because of how much memory its intermediate matrices consume. I then resorted to singular value decomposition to reduce the dimensionality of the data set. However none of this positively impacted accuracy
3. I created 'polariy_text' and 'subjectivity_text' columns which showed how positive and negative a review was and how subjective it was. These features did not help with accuracy either.

Model Selection

The final model chosen was Naive Bayes which resulted in the maximum accuracy I could achieve. I chose this model because it is designed for count based features, handles dimensionality well and is computationally very efficient. It is simple and can also handle class imbalances which were present in the given data. I initially used Multinomial Naive Bayes but then switched to Complement Naive Bayes which is specifically for an imbalance dataset. Doing so improved the accuracy. Final accuracy was 60%.



Other models explored: Some Other models I explored were 1) KNN 2) Random Forest 3) XGBoost 4) SVM