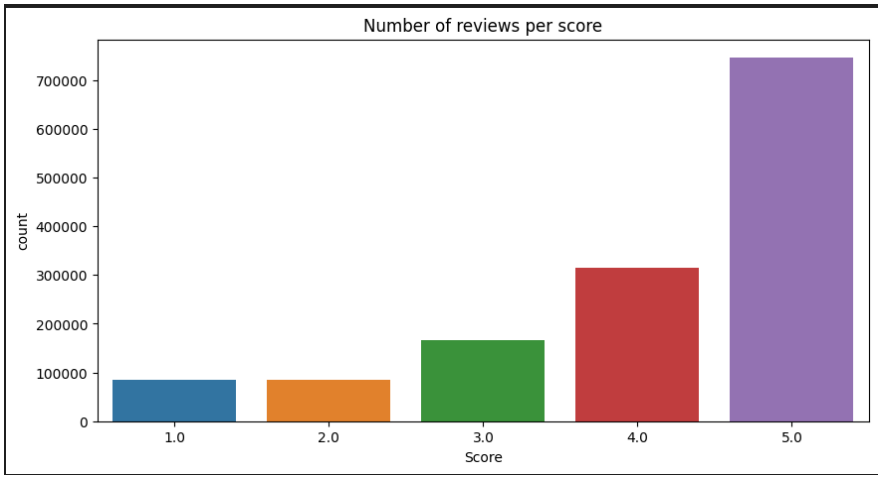
Report of Amazon Movie Rating prediction
Xiang Li
U53450247
Kaggle: Xiang Li

Preliminary analysis and exploration

There is a total of 1697533 rows of data with 9 properties(HelpfulnessDenominator, HelpfulnessNumerator, Id, ProductId, Score, Summary, Text, Time, UserId). The score column contains the rating score. The train.csv has many 300000 rows which miss the score information. The following graph plot the distribution of rating.



Feature Extraction

By the given start code, I create two separate data files. The X_train.csv file removes all the columns missing and the X_test.csv file is all the rows in which the ‘Sore’ value is missing. And add a column Helpfulnessto show the like rating rate. And I create the following features to help modeling.

1.Pop_summary: The popularity score of the Summary column, obtained by TexBlob sentiment analysis.

2. Pop_text: The popularity score of the Text column, obtained by TexBlob sentiment analysis.

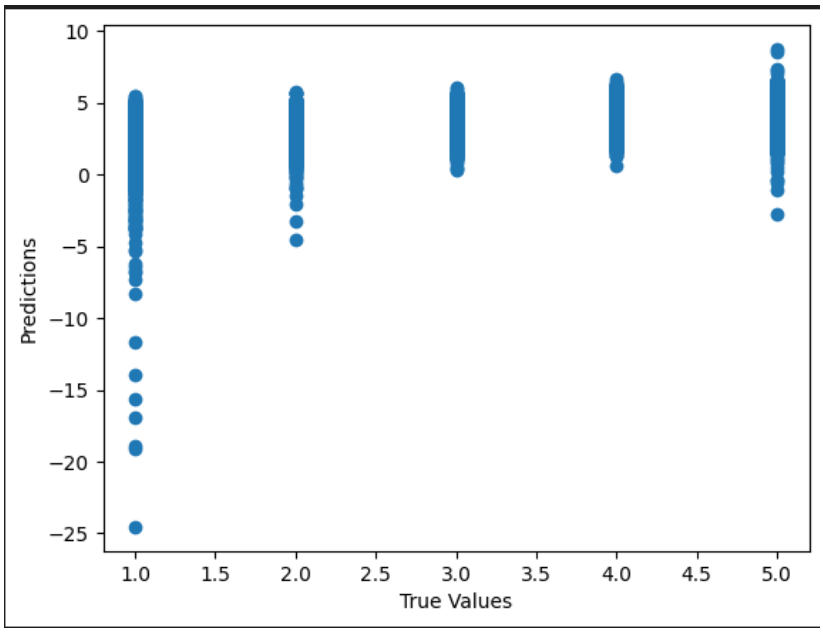
3. Total: TfidfVectorizer word frequency of the words in the features.

Workflow

After creating the X_trian.cvs and X_test by running the feature_extraction.py. I imply the following steps on both files. Stem every word in the text and summary column then use TfidfVectorizer to count the frequency of word features. Here I set the max_df=0.999 and mi_df=0.005. This is because if a word appears less than 0.5% of the whole data, it can be considered negligible. And if a word appears more than 99.9%, chances can be very high s.t. It had no specific mean.

Then I count the popularity of the Text and Summary and add them as a numeric feature to the original data

Then I contact the Tdif matrix and all the numeric features. By the train-test-split I split the contact dataset to 75% training and 25% test. And use the Linear regression model to get the prediction. After trying different models on the split data, I find the Linear Regression with test size= 0.25 shows the best RMSE. I will explain how I choose the model and test size in the Model part.



Model

I use the Linear Regression model with test_size=0.25 to predict the data, which gives me an RMSE of 0.91. I also tried test_size=0.4, test_size=0.3, and test_size=0.2 and they give me the RMSE of 0.93, 0.93, and 0.92.

I tried the Random Forest model by setting the n_estimator=200, which gives accuracy_accuracy of 56% and takes a very long running time which is not a huge improvement compared to the given KNN method.

I tried the SVM method, which gives me an RMSE of 0.97.

Based on the above information, I decide to use the Linear regression method.

Creativity, Challenges, and Effort

In the process of dealing with the text data, I found that the column size can be different after contacting the data together. This can result in an error when predicting the model. So I decided to add the Summary and Text columns together before running the TfidfVectorizer.

Most of the data used for training the model are from the Text and Summary columns.

One of the challenges I face is the memory error. It happens when my kernel can't allocate more Virtual Memory to process my program. The way I solve it is to save the big DataFrame to a csv file and restart the kernel and read this saved csv file.