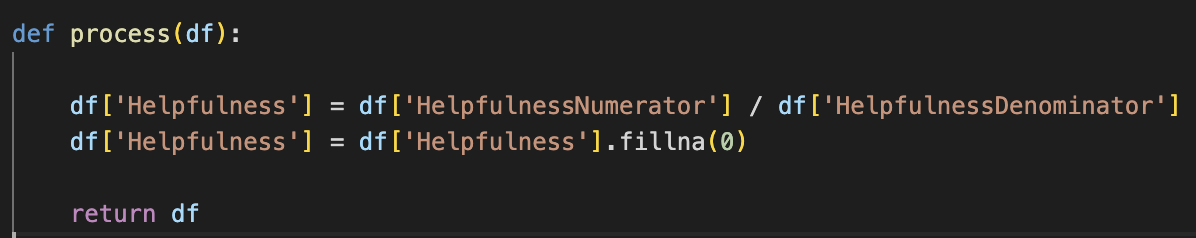
CS506 Midterm Report

Kaggle Username: Yiwei Zha

**Preliminary analysis / exploration / Preprocessing**

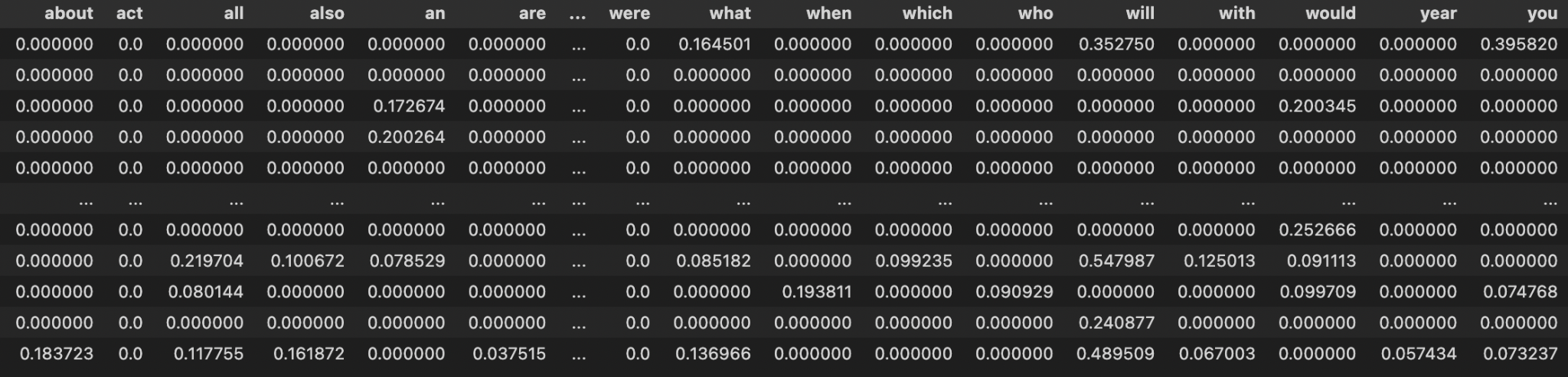
* HelpfulnessDenominator and HelpfulnessNumerator

At the start of examining the columns of the dataset, I found these two columns can be calculated into one column since the denominator is the number of likes and dislikes, and the numerator is the number of likes. Thus, I construct a new column of “Helpfulness” by writing a function below:



* Text and Summary

These two columns are non-numerical. However, the content of the text and summary can be the key to predict the review scores. Thus, by using SnowballStemmer, I first stem every word in these two columns. Then, by using TFIDF vectorizor, I make all the text into a numerical matrix showed like below:

****

In this way, I can further extract “Text” and “Summary” as features into model training which may greatly improve my prediction score

Beside the TFIDF vectorizor, I used a package called TextBlob, which is a Natural Language Processing package that did not use deep learning. By using this package, I was able to address a polarity score based on the word choice of “Text” and “Summary” columns. After using the TextBlob, I created two columns called “Text sentiment” and “Train sentiment”.

In this way, it also created numerical value for non-numerical columns which could used for building models.

**Feature extraction**

When extracting features, my consideration mainly focused on whether the feature gives a positive effect on the score prediction.

Originally, the training dataset has columns: “Id”, “ProductId”,”UserId”,”Time”,”HelpfulnessDenominator”,”HelpfulnessNumerator”.

However, some of the columns can be directly dropped which had no effect on prediction.

For instance, “Id” since Id number is just like the index of the reviews which is unrelated to the review scores.

Based on the same reason, I exclude “ProductId”,”UserId”,”Time” from the features.

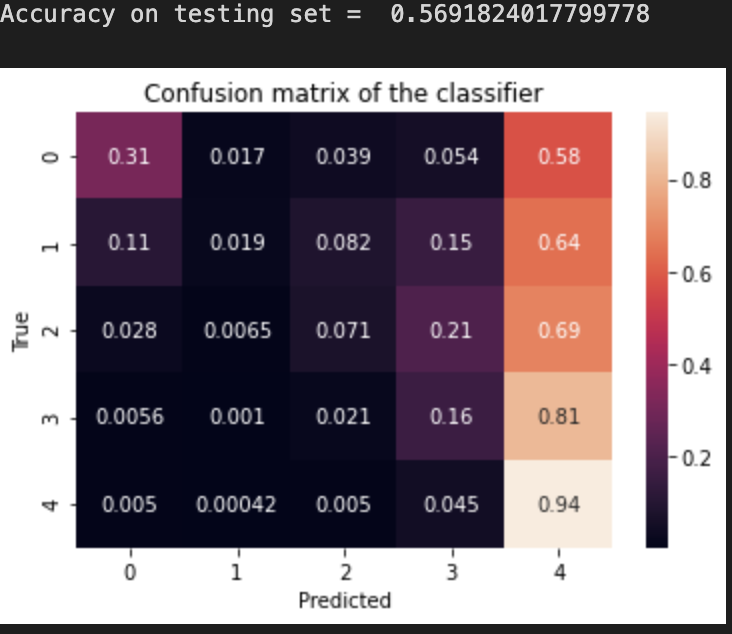
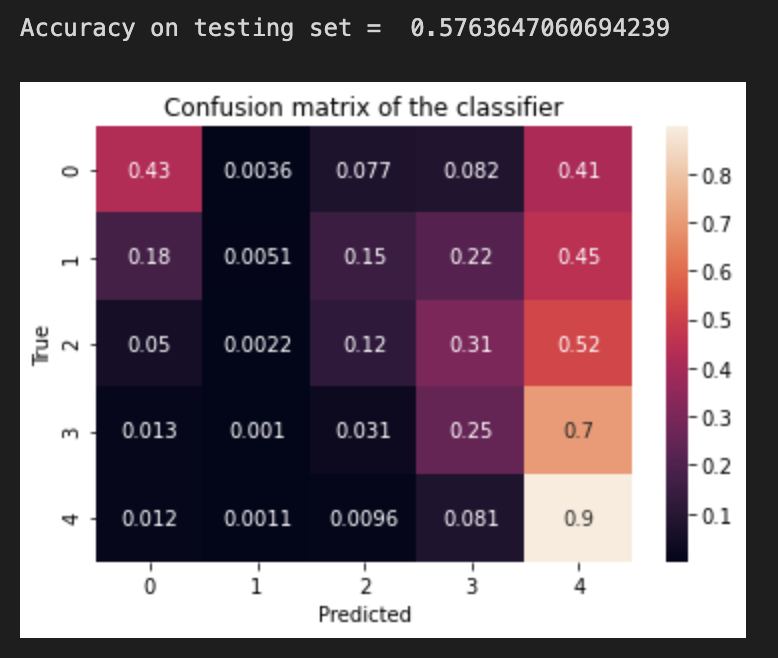
For Text and Summary, as our preliminary process, we have a TFIDF matrix, Text sentiment and Summary sentiment which all can be our features.

**The workflow, decisions, and techniques tried**

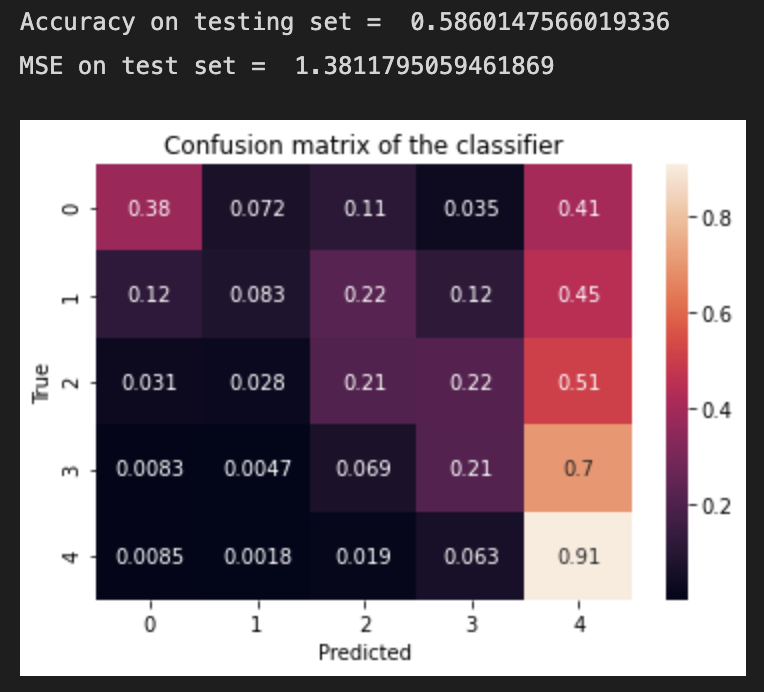
At first, I didn’t think of using the Textblob package. Thus, I just tried to use Random Forest, SVM, Logistic Regression, Linear Regression these methods to fit the model with HelpfulnessNumerator, HelpfulnessDenominator, Helpfulness and Text/Summary TFIDF matrix.

However, all these methods based on the same features didn’t have a really small mean squared error or a high accuracy score.

* SVM - Random Forest

****

* Logistic Regression



* Linear Regression



The former is the R square and the later is the MSE.

After that, I processed the Textblob package on Text and Summary. I added the Text sentiment and Summary sentiment to the features and used the same method again. In this way, I found my best estimation was to use linear regression with features of “Helpfulness”,”HelpfulnessNumerator”, “HelpfulnessDenominator”,”Text\_sentiment”,”Summary\_sentiment” and the TFIDF matrix.

**Model tuning/testing**

* Try to add more features

After finding my best model for predicting, I stuck on how to improve on it. I tried to add more features like “Time” on to the dataset to see whether the model prediction improved while it didn’t.

* Try to use more data

For every model, I use the whole dataset to fit the model so that I have no chance to improve the model by extracting more data

* Try to tune on the feature construction

For TFIDF matrix, previously I set the max\_df to be 0.85 and the min\_df to be 0.15. In this case, the model MSE is 0.904. While I reset the max\_df to be 0.999 and the min\_df to be 0.05, and add on the max\_features to be 10000, the model MSE is 0.826. I thought the reason why the model improved was that the original TFIDF matrix based on the default settings didn’t include all the words in the Text and Summary. However, if we add on the max\_features to be 10000 which is an unreachable limit, the new TFIDF matrix includes all the words as columns.

**Challenges/effort**

* Merging of Datasets went wrong

Originally, I didn’t put the Text and Summary column together as a new column ‘Total’. Thus, I did the TFIDF matrix on both columns. However, when I tried to merge them together, I found no matter what variable I set, I could not merge them into a new dataset that had the same row number. The new merged dataset had twice the row number as the original had. And many values were NaN. Thus, I thought that we could put the Text and Summary column together as a new column ‘Total’ since the TFIDF matrix will become a big matrix which include both of them. In this way, I didn’t encounter this error again and make the dataset concatenation well.

* Chaotic notebook arrangement

Since running the word stemmer and the TFIDF took lots of time, I didn’t arrange my cell properly and save them into csv so that I could read them next time. I spent lots of time rerun the code.

Also, due to the various machine learning method I used, there were a bunch of different variables addressing the same dataset. Sometimes, I could not figure out which dataset was modified by me which wasn’t. This troubled me hard during the model training.

Hence, I rewrote all the code in a new notebook so that I could run the code with no mess.

* Machine Learning Method Self Learning

I used many machine learning methods that we hadn’t taught. This was because I was eager to learn new methods on practical use. I believe, in practical, I could learn them well.