Midterm Report

Preliminary analysis / exploration

- Analyze the sample number of each score
  - The scores of more than 75% samples are above or equal to 4, and the scores of more than 50% samples are equal to 5.
- Analyze the sample number of each product / user
  - There are 50052 different products, and the training set covers 50050. So, the submission set will face two new products.
  - There are 1239060 different users, and the training set covers 123958. The submission set will face two new users.
  - The distribution of product and user prevents the method that analyze the product and user separately.

Feature extraction

- Encode the Id of product and user to Integer
  - Reason:
    - The Id is object value, which cannot used in model
    - Product and user may also be the important feature of samples
  - Method: pandas.factorize function
- Normalization
  - Reason: Some columns have large value that might influence the final result
  - Method: Z-score: x-x.mean()/x.std()
- Convert the summary / text of each sample to numeric
  - Reason: The summary and text are the best way to show the user's attitude to the product. But they are object value.
  - Extracted the features in summary and text separately, saved as *_summary.csv, *_text.csv
  - If set TdifVectorize threshold from 0.1 to 08. There are only three columns based on vocabulary in summary. It means the summary contains little information. So, focused on text.
  - Converted the text to same length vector because the order of words is also important. Saved as *_textVector.csv.
- Calculated the length of summary and text

Workflow, decisions, and techniques

For every model, the program will read csv, select useful columns, split to train and test dataset,

train model, predict and validate by accuracy and mean absolute error, and finally, predict on submission set and save.

The data has score, which is target. So, it is supervised learning. Chose K-nearest Neighbor, Random Forest, Support Vector Machine, and Logistic Regression to predict.

Model tuning / testing

K-nearest Neighbor

- Applied cross validation on KNN to look for the best k.
- But it performed bad
  - The running time is really long because of the high dimension data and cross validation. So, I tried to reduce the dimension by add all positive words to a new column 'Positive' and reduce the number of cv.
  - The confusion metric shows that most of predictions are 5. Due to the high running time cost to try it again, I move forward to other algorithms.

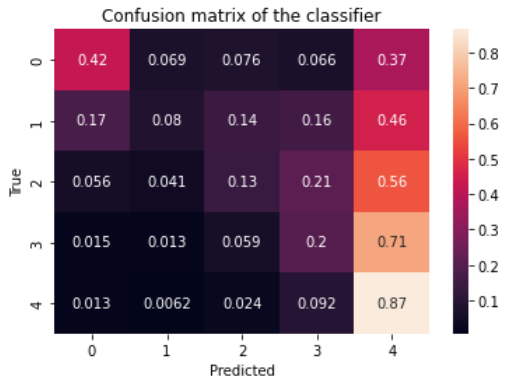
Support Vector Machine

It has the same problem with KNN. High running time cost push me to the next algorithm.

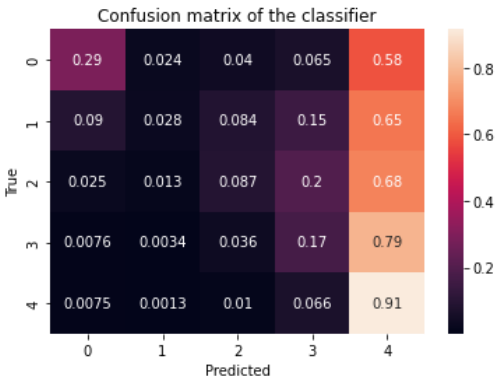
Random Forest

- The predictions of random forest model are better than that of KNN and logistic regression. It achieved lowest MSE and its running time was not long.
- Tunning focused on the number of trees and max depth. To avoid high running time cost, I only validated the model on test set, and didn't use cross validation method.
  - Tried the number of trees from 5 to 101, step is 5. After get the optimal tree number, look for optimal max depth from 30 to 101, step is 5.
- Figure:

cm after tunning parameters



cm using text vector

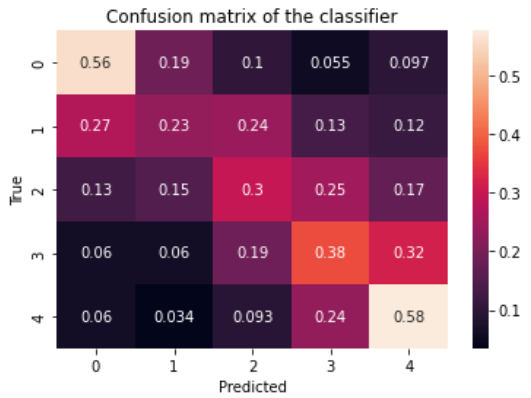


- The two figures above show the final result of the random forest.
  - Model after tunning parameters performed similar as before, but it lost some accuracy to reach smaller mean squared error. But in conclusion, tunning parameter improved the model a little.
  - Model fitted by sentence vector and after tunning performs worse than the other one. The order of the word is important, but converted the sentences to same length vector increased the noise in data.

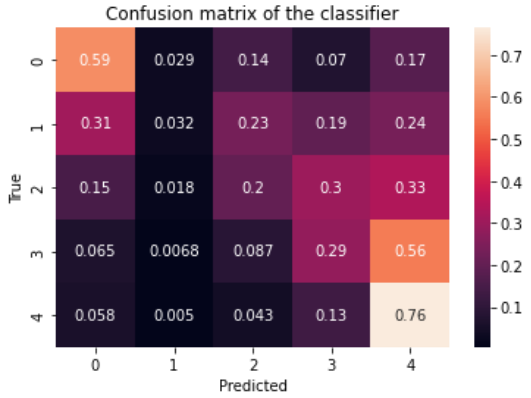
Logistic Regression

- Balanced and unbalanced logistic regression
  - The logistic regression predictions are similar to that of KNN, if the weights of classes are default
  - If set the weight of classes based on dataset, although the accuracy and MSE are worse than default, the confusion metric shows that the model found some actual differences among these scores.
- Logistic Regression Group
  - It is aimed to building voting group to predict the score
  - Built 6 regression models. The first one was fitted by all samples, which is a multiclass regression. And the other five were fitted by samples of one score and same number of other samples after setting the score to 0, which is five binary regression models.
  - The confusion metric show that the predictions of score 2 are similar to the single regression. And the accuracy of score 5 decreased a little, which increased the accuracy of score 1, 3, 4. And the total MSE and accuracy is a little better than set class weight parameter.
- Figure:

cm of set class weight param



cm of regression group



- In conclusion, the logistic regression with all text features, text length and default class weight achieved the best score. The result is shown in 'regression_text.ipynb'. But I think it is a coincidence, because there is no universality. The data has many nosie and performed bad on random forest.
- Besides, I prefer the regression model with some keywords, and set the class weight based on the dataset. It performed stable. The result from knn, random forest and logistic regression are similar.