**CMPE 256-01 Programming Assignment – Book Recommendation System**

**Submitted By: Kashika Jain SID: 012505649 Email:** [**kashika.jain@sjsu.edu**](mailto:kashika.jain@sjsu.edu)

**Rank – 9 RMSE – 1.5577 Kaggle ID - Kashika**

**Objective**

The objective of this assignment is to predict the rating that a user will give to a book given their past book rating using recommender system algorithms.

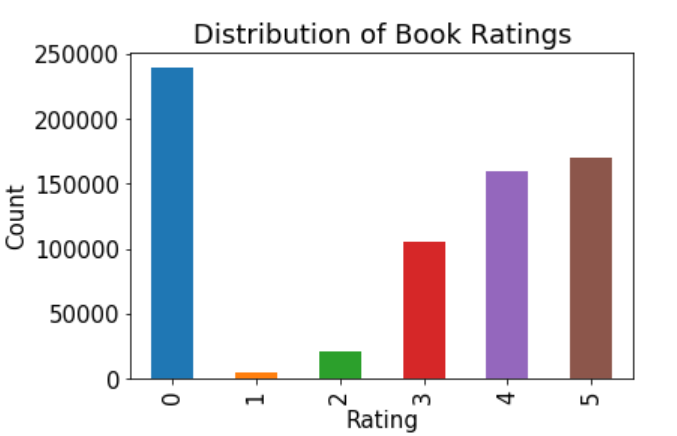
**Dataset**

The dataset has 700K training data about how users have rated books in the past and 300K test data to test our recommendation system model.

**Methodology -**

**Data Cleaning and Preprocessing**

1. The first step performed is to load the data and import the libraries that we will use to build the recommendation system. In this assignment I have used surprise library.
2. I performed some exploratory data analysis to understand how the data is distributed. The first step was to understand how the ratings are distributed in the training dataset.



1. The next step I performed was to understand the popular books that the users rated more by grouping the data based on book\_id and their ratings.

4. The reader class in Surprise is used to parse a file containing ratings and such a file is assumed to specify only one rating per line. I have converted our dataset to surprise reader dataset.

5. The next step is to perform the test train split of data as we need to check our RMSE in the validation data set to understand how the different algorithms are implemented. We use the split size as 0.25. In some cases, we also use cross validation

**Algorithms Implemented**

1. **SVD** – I have implemented SVD which is Probability Matrix Factorization technique to perform recommendation system based on user-item-rating utility matrix. The RMSE for validation data was 1.6081 and test data was 1.6289

2. **KNNBasic-** Next I implemented KNNBasic which is Collaborative filtering technique to perform recommendation system. The RMSE for validation data was 1.8077 which was worse than our previous approach

3. **GridSearchCV -SVD –** Grid search cross validation is used to perform an algorithm with different parameters like to get the best parameter value to fit our dataset. I passed different values to various parameters to check the best RMSE for 3-fold cross validation. I received an RMSE of 1.5873 which is an improvement from above 2 cases. Later I performed more detailed parameter tuning as described below-

'n\_epochs': [10,15,20,30,40,50,60,70], where n\_epochs is the number of iterations

'lr\_all': [0.005, 0.001, 0.5, 0.1, 0.05], where lr\_all is range of linear regression parameter values

'reg\_all': [0.1,0.5,0.01,0.005,0.4], where reg\_All is the range of regression parameter range

4. **Baseline Approach –** Next, I tried using baseline approach which is an algorithm predicting the baseline estimate for given user and item. The algorithm gave very good RMSE score on validation set which was 0.419, however performed badly in test data which was 2.15 because of overfitting.

5. **SVDpp-** To experiment further, I implemented SVDpp which is an extension of SVD considering the implicit ratings. The algorithm performed well in validation set giving an RMSE of 0.64 but gave RMSE of 2.14 in test set.

6. **Other Techniques -** Some other techniques that I implemented included NMF (Non-Negative matrix Factorization) and many other combinations of SVD using grid search algorithm. The best outcome came with SVD using grid search and using 10-fold cross validation. The test set gave RMSE of 1.55

**Comparison of algorithms**

**Conclusion**- After implementing the various recommendation system algorithms as described in the report above, I noticed that the best algorithm was SVD with fine tuned parameters using grid search algorithm. The best parameters were {'lr\_all': 0.005, 'n\_epochs': 60, 'reg\_all': 0.1} using 10-fold cross validation giving a test RMSE of 1.557 as can be seen in the chart above.

**References-**

1. Class Demo by Niraj

2. <https://surprise.readthedocs.io/en/stable/getting_started.html>

3. <https://surprise.readthedocs.io/en/stable/basic_algorithms.html>

4. <http://surpriselib.com/>

5. <https://surprise.readthedocs.io/en/stable/matrix_factorization.html>