**East West University**

**Course Title: Big Data Analytics**

**Course code: CSE488**

**Fall2023**

**Project Report**

**Submitted To:**

**Dr. Mohammad Rezwanul Huq**

Associate Professor

Department of Computer Science and Engineering

East West University, Dhaka

**Submitted By:**

|  |  |
| --- | --- |
| **Gazi Fayaz Ahmed** | **2020-1-60-099** |

Department of Computer Science and Engineering

East West University, Dhaka

Date of Submission: 29/ 12 / 2023

**Task 1**

**Link:** [**https://colab.research.google.com/drive/1E8flkO9OBCqshSrDxbHtLSpKOfm7k8xu?usp=sharing**](https://colab.research.google.com/drive/1E8flkO9OBCqshSrDxbHtLSpKOfm7k8xu?usp=sharing)

Significant Steps:

1. Remove the rows with the rating ‘0’ from the preprocessed dataset
2. Include only the users with at least more than 30 rating counts
3. From that, only include the books with at least 70 rating counts
4. Make User-Book Matrix using pivot table
5. Subtract row mean from every row and store it as a normalized matrix
6. Fill the NaN values with 0
7. Generate Cosine Similarity of every user with other users
8. Define a function to get tuples of user index and similarity value with other users for a specific user
9. **Generate predictions for all the books with no ratings or ‘0’ as rating for every user based on the weighted average of the ratings of the first 3 most similar users who have rated the book.**

**Output Sample:**



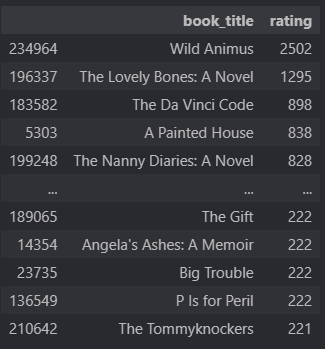
**Task 2**

**Link:** <https://colab.research.google.com/drive/1XKxkR_Ws-WpWhTUz_y3JphclGmGfq_BM?usp=sharing>

Significant Steps:

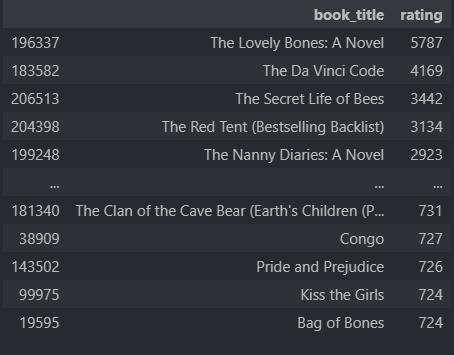
1. Find the number of ratings for each book
2. Find the threshold value for top 0.1% of rating counts
3. **Find the top 0.1% of books in terms of rating counts.**

**Sample Output:**



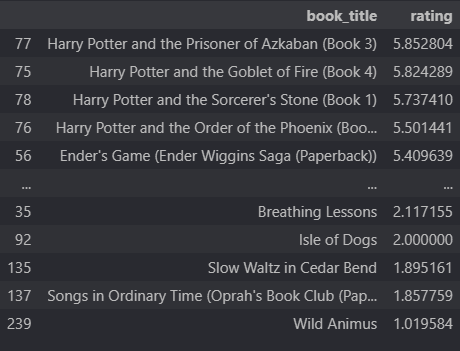
1. Find the sum of all ratings for each book
2. Find the threshold value for top 0.1% of rating sums
3. **Find the top 0.1% of books in terms total rating.**

**Sample Output:**



1. **Find the average ratings of the top 0.1% of most times rated books.**

**Sample Output:**

****

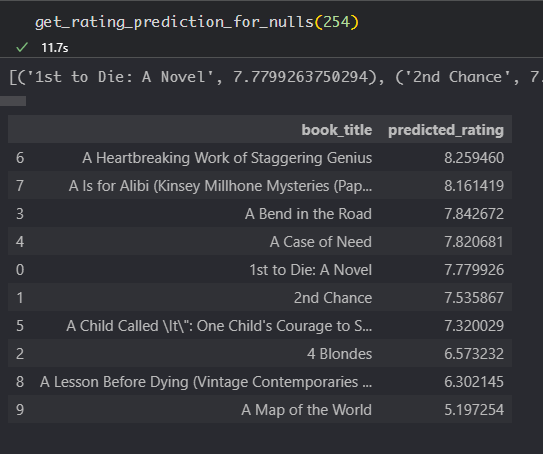
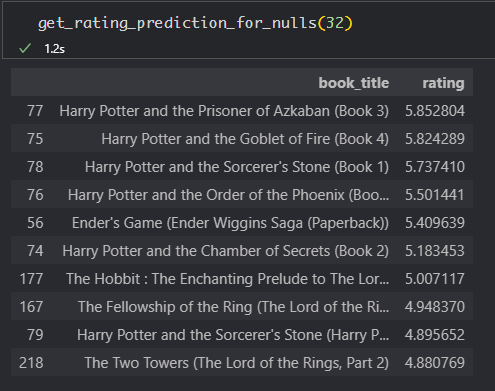
**Task 3**

**Link:** [**https://colab.research.google.com/drive/1ij\_5RA88LPGQlIK8pDbK5srzZeCZz9oi?usp=sharing**](https://colab.research.google.com/drive/1ij_5RA88LPGQlIK8pDbK5srzZeCZz9oi?usp=sharing)

Significant Steps:

1. Include only the users with at least more than 50 rating counts
2. From that, only include the books with at least 100 rating counts
3. Make User-Book Matrix using pivot table
4. Subtract row mean from every row and store it as a normalized matrix
5. Fill the NaN values with 0
6. Generate Cosine Similarity of every user with other users
7. Define a function to get tuples of user index and similarity value with other users for a specific user
8. **Define a function that takes user id. If that user id is present in the matrix, then predict rating for each book that the user has not already read and based on the prediction ratings, recommend the top 10 rated books. Otherwise, recommend 10 most popular books in terms of most times rated and sorted by average ratings.**

**Sample Output:**

**** ****

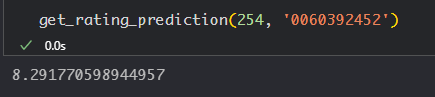
**Task 4**

**Link:** [**https://colab.research.google.com/drive/1S9hZ8ElIaxfaJDFlmlyGo\_\_jKmm5Jd2b?usp=sharing**](https://colab.research.google.com/drive/1S9hZ8ElIaxfaJDFlmlyGo__jKmm5Jd2b?usp=sharing)

Significant Steps:

1. Remove the rows with the rating ‘0’ from the preprocessed dataset
2. Include only the users with at least more than 10 rating counts
3. From that, only include the ISBNs with at least 50 rating counts
4. Make User-ISBN Matrix using pivot table
5. Subtract row mean from every row and store it as a normalized matrix
6. Fill the NaN values with 0
7. Generate Cosine Similarity of every user with other users
8. Define a function to get tuples of user index and similarity value with other users for a specific user
9. **Define a function that takes user id and ISBN value to predict a probable rating by checking the 3 most similar users who have rated the same book and weighted averaging their ratings**

**Output Sample:**

****