

Department of Computer Science and Engineering

Fall-2023

Course Code: CSE488

Course Title: Big Data Analytics

Project Title: Book Recommender System

Colab link: [Click here](https://colab.research.google.com/drive/1815mdDPUpc8z9GoQiOaxKibThn3Hl7Kn?usp=sharing)

**Submitted from:**

Name: Abdullah al Tamim

ID: 2020-1-60-127

**Instructor:**

Dr. Mohammad Rezwanul Huq

Associate Professor, Department of CSE

East West University, Dhaka

**Project Description**

The Book Recommender System is designed to suggest new books to users by identifying shared preferences among them. I was assigned the task of implementing collaborative filtering using the K Nearest Neighbor approach to cluster users based on common book ratings. This involved utilizing the average rating from the top k nearest neighbors to predict outcomes. The project focuses on two main functions: the first takes a user ID as input, providing a list of the top ten recommended books for that user, while the second function, taking a user ID and ISBN number as input, predicts the probable rating for the specified book. The goal is to tailoring book recommendations and predicting potential ratings.

**Dataset Description**

The dataset that I have used was named as ‘Preprocessed\_data.csv’. It was obtained from the provided link in the classroom. Initially the dataset had 19 columns and 1031175 rows. The dataset contains information about books, users and ratings given for the books by the users.

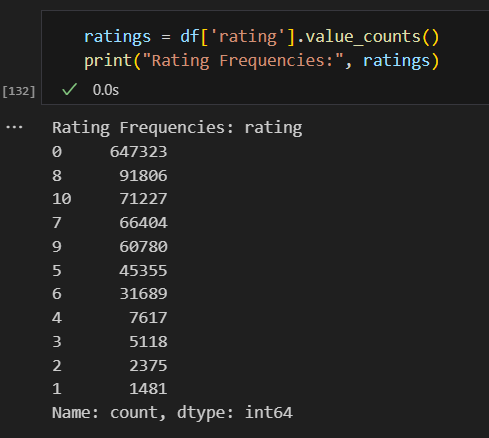
**Exploratory Data Analysis (EDA)**

* At the beginning of EDA part I dropped some unnecessary columns specifically irrelevant for our given tasks. Which are: ['img\_s', 'img\_m', 'img\_l', 'Summary', 'city', 'state', 'country'].
* Fortunately, there were no missing values in the remaining columns.
* The number of unique values for each column is shown below:

A screen shot of a computer

Description automatically generated

* Then I tried to identify the distribution of each type of rating:



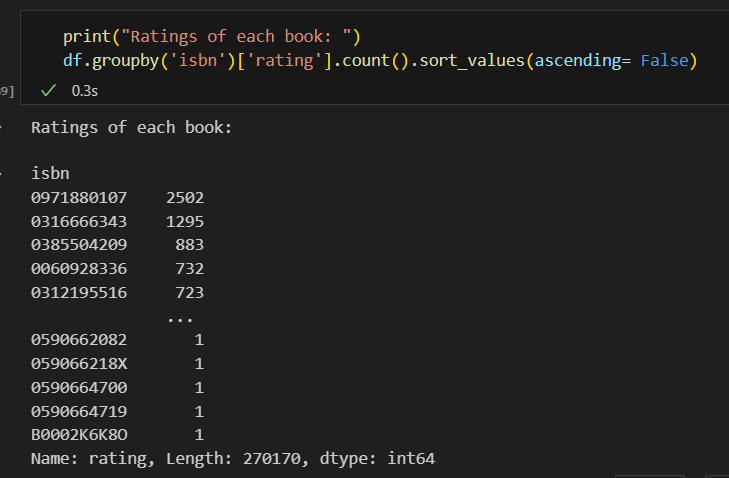
And found out that 647323 users didn’t gave any rating at all, those ratings are shown as 0.

* Then I have tried to find out number of ratings given by each user:

A screenshot of a computer

Description automatically generated

* In the same way I figured out number of ratings for each book:



**Building the User-Book Matrix**

Finally I made a new dataframe with those users who have rated more than 100 books and with those books which has got more than 100 ratings. And from that dataframe, I made a matrix where rows are indexed with user ids, columns with isbns and values are the rating. So a cell of that matrix, mat[x,i] contains the rating of the book i given by user x.

**Intuition of Finding Pearson Correlation Coefficient**

The Pearson Correlation Coefficient function that has been used throughout the project isn’t any library function. I recreated the function by myself. The main idea was first of all subtracting the mean ratings of each user from their each rating. And then calculating the numerator, which is calculating the sum of multiplication of all the common ratings between the input user and the other users individually. And Finally the denominator which is the multiplication of normalization value of two users.

***Pseudocode:***

def pearson\_correlation\_coefficient(user\_book\_matrix, user\_id)

user\_book\_matrix = user\_book\_matrix – user\_book\_matrix.rows.mean()

for other\_user\_id, other\_user\_rating in user\_book\_matrix:

common\_ratings = user\_book\_matrix[user\_id] & user\_book\_mat[other\_user\_id]

numerator = user\_ratings\*other\_user\_ratings

denominator = sqrt(sum(user\_rating)\*\*2)\* sqrt(sum(other\_user\_rating)\*\*2)

correlation\_df.append(user\_id, other\_user\_id, numerator/denominator)

return correlation\_df

**Intuition of the First Function**

To complete the first task, i.e. a user\_id will be given as input and we have to find out the top ten recommended book for that user. To do so, initially I measured the similarity of the input user with all the other users using Pearson Correlation Coefficient. Then based on the similarity, I have made a cluster with 10\*3=30 users. From that user cluster I have formed a list of recommendable books excluding the books that the input user has already rated. Then I suggested top ten books from that list based on the weighted average of that book given by the users of the cluster.

***Pseudocode:***

def recommend\_10\_books(user\_id, correlations)

user\_cluster = correlations.nlargest(n\*3, “correlation”)

books\_recommendable = get\_recommendable\_books(user\_cluster, user\_id)-books\_readed

top\_10\_books = books\_recommendable.nlargest(n, “weighted\_avg\_rating”)

return top\_10\_books

**Intuition of the Second Function**

The second task was that a user id and isbn number will be given and we have to predict the probable rating of that book given by that user. To do so I have used user-user collaborative filtering approach. First of all, I measured the correlation of the user with only those users who has rated that book. Finally, I predicted the probable rating using baseline estimation, weighted average and only average of the ratings of top correlated users.

***Pseudocode (Baseline Estimation):***

def predict\_rating\_bestimate(user\_id, isbn, filtered\_user\_book\_mat, correlations)

top\_correlations = correlations.nlargest(n, “correlation”)

for rating in filtered\_user\_book\_mat[top\_correlation[‘other\_user\_id][isbn]:

ryi\_byi.append(rating – get\_baseline(filtered\_user\_book\_mat, other\_user\_id, isbn)

numerator = top\_correlations[‘correlation’]\*ryi\_byi

denominator = sum(top\_correlations[‘correlation’])

predicted\_rating = get\_baseline(filtered\_user\_book\_mat, user\_id, isbn) + numerator/denominator

return predicted\_rating

***Pseudocode (Weighted Average):***

def predict\_rating\_weightedavg(user\_id, isbn, filtered\_user\_book\_mat, correlations)

top\_correlations = correlations.nlargest(n, “correlation”)

numerator = top\_correlations[‘correlation’]\*filterd\_user\_book\_mat[user\_id,isbn]

denominator = sum(top\_correlations[‘correlation’])

predicted\_rating = numerator/denominator

return predicted\_rating

**Conclusion**

In conclusion, the exploratory data analysis (EDA) for our Book Recommender System has provided valuable insights into the dataset, paving the way for informed decision-making. I have successfully implemented collaborative filtering using the K Nearest Neighbor approach, revealing user clusters based on common book ratings and predicting outcomes through the average rating of the top k nearest neighbors. From the project we have got a brief knowledge about recommendation systems. We can implement this knowledge in websites or applications now.