PROJECT SYNOPSIS
ON

‘BOOK RECMMENDATION SYSTEM FOR NEW READERS’

SUBMITTED IN PARTIAL FULFILMENT OF THE
REQUIREMENT OF UNIVERSITY OF MUMBAI FOR THE
DEGREE OF

MASTER OF SCIENCE IN INFORMATION TECHNOLOGY

SUBMITTED BY
SADIQ MOHD ALAM SHAIKH
ROLL NO. - 6928

UNDER THE GUIDANCE OF
PROF. ABHIJEET SALVI

DEPARTMENT OF INFORMATION TECHNOLOGY

PILLAI COLLEGE OF ARTS, COMMERCE AND SCIENCE, NEW
PANVEL – 410206
UNIVERSITY OF MUMBAI

ACADEMIC YEAR: **2020 – 2021**

DECLARATION

We declare that this written submission for a Project Synopsis Declaration entitled "**BOOK RECOMMENDATION SYSTEM FOR NEW READERS**" represent our ideas in our own words and where others' ideas or words have been included. We have adequately cited and referenced the original sources. We also declared that we have adhere to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any ideas / data / fact / source in our submission. We understand that any violation of the above will cause for disciplinary action by institute and also evoke penal action from the sources which have thus not been properly cited or from whom paper permission have not been taken when needed.

Project Group Member(s):

Name of student: _________________

Date:

Place:

CONTENTS

LIST OF FIGURES

ABSTRACT

Recommendations are an important part of various ecommerce and seller websites. Recommending a product to the users has helped a lot of these companies in predicting the interests on that product. A Book Recommendation System can be helpful to many people as they will be suggested what book to read next. These can be learned from the user's choices over time and a 'taste' profile can be created by it. In recent years, with the improvement of computing level, it is possible to provide personalized recommendation lists for users based on user behavior. Increasing number of books in library in university has an urgent demand for personalized book recommendation. Existing approaches for Recommending involves selecting from two of the popular approaches i.e., Collaborative learning and Content Based learning. A model is established by author using collaborative filtering algorithm, and targeting students who have never borrowed books from the library. The model generates the book recommendation lists for the target users by using their course selection records and existing borrowing data of known users.

Keywords: Collaborative Filtering Algorithm, Web Scrapping, Book Recommendation.

CHAPTER 1

INTRODUCTION

What is Recommendation System?

There is a large growth in the amount of data available on the internet coupled with the fact that there are a large number of internet users as well. This in turn has led to the problem of exposure to this huge data leading to information overloading. Many IR systems i.e., Informal Retrieval system have tried to solve this problem. They have partially solved the problem but still the large problem lingers on which is known as personalization. Personalization in simple words means tailored for a specific user or a group of users according to his/her interests. Hence for this very purpose of personalization or customization, Recommender systems came into the picture. Recommender system is basically a filtering system customized according to the user's needs.

BACKGROUND OF RECOMMENDATION SYSTEM

The concept of recommendation system was first proposed by the United States in the 1990s and it was first applied in the commercial field for increasing sales by digging out the huge long tail goods. The recommendation system has currently spread around every corner of our life. For example, Taobao or Jindong will recommend goods or push discount information according to user's previous shopping and browsing behavior; toutiao.com will generate different news lists for different users based on previously viewed news. In the field of university recommendation, the most common recommendation system is based on thestudent library records to produce book recommendation list by the neighbor's collaborativefiltering algorithm.

However, the recommendation systems library based on students' borrowing records are unable to provide personalized book recommendations for new users who have never borrowed book in the library. Thus, this paper proposed a book recommendation model combined with students' course selection data and collaborative filtering algorithm to solve the cold start problem that the target user does not borrow records in the recommendation system. Recommendation in simple words means tailored for a specific user or a group of users according to his/her interests. Hence for this very purpose of personalization or customization, Recommender systems came into the picture.

RELATED WORK

There have been many researches on recommendation systems but there are few studies of book recommendations based on library automation with the limitation of not too much learning data such as library loan records, library category or title of books. Harada (2009) and Harada & Masuda (2010) used collaborative filtering. Tsuji et al. (2011) (2012) used 1,854,345 loan records from 39,442 users of the T University Library and recommended books to 33 undergraduate and graduate students based on the collaborative filtering method that was proposed by Harada & Masuda (2010). Many approaches rely on collaborative filtering (CF) methods based on the main idea that people have similar preferences and interests, so similarities of users or books are calculated. Basically, each user gives rating scores for a list of items and these scores are used to predict the rating active user. They found that the evaluations of these methods were ranked from best to worst as Amazon, association rules, and then collaborative filtering.

Another approach of recommender system is based on data mining techniques such as association rule, clustering, and decision tree. A library book recommendation system based on user profile loaning and apply association rule to create model was proposed by. Association of users, book categories, and book titles are explored to extract rules. In, features of classification, user based collaborative filtering, and association rule mining are combined to develop the technique which recommends most suitable books to the students according to their price range and publisher's name.

Due to rapid growth of data, sparsity of rating matrix has become a challenge issue in recommender system. Many researchers have adopted matrix factorization techniques to deal with sparsity problem. The experiment on dataset such as Netflix Prize data has shown that they are superior to classic nearest neighbor techniques. In, missing values of MovieLens dataset were filled and then matrix factorization model SVD was applied to reduce the dimensionality of a rating matrix. On the other hand, proposed the system named Eigenstate which use PCA to address sparseness

Objectives

The primary objective of the Book Recommendation System is that will be suggested what book to read next. These can be learned from the user's choices over time and a 'taste' profile can be created by it. The system can also help ne reader to guide as to what book the reader to choose so that without wasting anytime the user can pick the right book on the required which he mentions in his prerequisites. Also, the students who are studying in Universities or colleges who need different who books to study for a particular subject will also get benefitted by this. The reading lover could also go with the recommendation of this system as it would also provide the reason why the particular book is recommended to the person.

Scope

This paper focuses on the problem of cold start, and tries to carry out personalized book recommendation with students' course selection, which is necessary for students during their college time. The main target user are the students of Universities or Colleges. Secondly, known users who are the most similar to the target user are selected (neighbors) to calculate the recommendation coefficient for each book according to the book borrowing record of known users, then the personalized book recommendation list for the target user will be generated.

Outline

The report is organized as follows: The introduction is given in Chapter 1. It describes the fundamental terms used in this project. It motivates to study and understand the different techniques used in this work. This chapter also presents the outline of the objective of the report. The Chapter 2 describes the review of the relevant various techniques in the Literature Systems. It describes the views of different authors as to what they have researched. It includes pros and cons, technique, accuracy of each Research Paper. The Chapter 3 presents the Proposed work. It describes the major approaches used in this work. In the Chapter 4, The flow and the details of the proposed and technical applications are mentioned. The summary of the report is presented in Chapter 5.

CHAPTER 2

LITERATURE SURVEY

Based on the rapid development of e-commerce platforms and the increasing technology and Internet availability in extra-curricular activities, many platforms with book recommending have sprung up. Moreover, it also important to have a look on the research and throw light on the area and to look into how the existing models exists and what cons do they include and where they lag. Recommendation system used in the area of university library is aimed to solve the following questions. Hence, it is necessary to look up in the areas where people have their existing work and model with utmost level of accuracy. A survey is required before developing the proposed system. Acquiring massively and processing of all the techniques has triggered numerous research studies aiming to profiling human beings computationally and accurately.

According to the author, **Rohit Darekar, Karan Dayma, Rohan Parabh, Prof. Swapnali Kurhade** which belongs to Department of Information Technology, Sardar Patel Institute of Technology, Andheri, Mumbai, who made a research on Book Recommendation System using Collaborative based filtering (CF) and Content based filtering (CB) which had a good accuracy to recommend books for the reader on a particular platform. They proposing a novel hybrid book recommender system combining the different recommender algorithms to optimize our recommendations. The system will find out which book the user had bought earlier i.e., the genre or category of the book and then finding out the sub-category if it could have. Performing filtering to find out the list of similar books and from the book transaction database find the transactions whose category as well as subcategory.

In the system proposed by them, there is a Login Module through which the user logins the system. The login module would be connected to a database where the user data i.e., profile will be stored. It will also include the books viewed and rated and also other information of the book. On logging into our system, the user will be able to view all the books and even rate them. The proposed recommender system will take into account result of the union of both the approaches i.e., collaborative as well as the content-based filtering. They also took help of demographic filtering approach if we do not get results from the above two approaches.

Observations: The recommendation system best fits the user's perspective but lags gaining the changing interests on the people. The main problem here is that user always don't have same thing to watch or same thing to buy, i.e., the choice of users keeps on changing from time to time. An example of that is on one day user browse Flipkart for a book, but on another day the user might be searching for a book of some different genre of books after some day's user might like to read books of new genre but there are chances that the books suggested might be of different type of genre. This is one of the major problems concerning the recommendation system. There is large amount of data required to produce effective and accurate recommendations.

According to the author, **Wenyu Li, Daqiang Chen, Xiaoyu Duan, Changchang Huang, Yayun Lu, Xuemei Hu** School of Management and E-Business Zhejiang Gongshang University Hangzhou, China, the book recommendation system is designed to be prominent in accuracy, specialization, efficiency, safety and user-friendliness, etc. It is an effective and efficient design approach to adopt the modularized functional design to have the overview of the functions of book recommendation system. Using database, the E-R model of this book recommendation system has three entities. There are three essential tables comprising the database of book recommendation system accordingly, user information tables, which are used to record the essential information of users for matching the requirements.

The system includes modules - Platform Login Interface, Information Input of Users, The Main Interface of the Book Recommendation System, The Book Classification and Rating Interface, Book Management Function. The E-R model of this book recommendation system has three entities. There are three essential tables comprising the database of book recommendation system accordingly, user information tables, which are used to record the essential information of users for matching the requirements. The book recommendation system is designed to be prominent in accuracy, specialization, efficiency, safety and user friendliness, etc. It is an effective and efficient design approach to adopt the modularized functional design to have the overview of the functions of book recommendation system.

Operations: The system includes great accuracy to find the most recommended books but the system lacks the working and mechanism using the Artificial Intelligence and Machine Learning Algorithms Also, the system has certain limitations i.e., to update the database frequently and that too on a manual basis. The system doesn't use any of the latest framework to make a smart way to implement the process.

According to the author**, Qi Ji, Song Yannan, Liu Shi, Liu Xiang** School of Computer Inner Mongolia University, the proposed system, researcher conducted book recommendation using graph database and book metadata that is stored in Neo4j. Neo4j itself is a graph database allowing storing data as nodes that connected by arcs. Because metadata can be easily represented as graph. Neo4j provides a cypher query to extract data from graph database. Besides, there is also a research that creating Personalized Movie Database System (PMDS). It is a dynamic web application created for purpose to see information about movie. PMDS uses PHP as web programming and MySQL database as relational database.

Main criteria for good database are respond time and the result itself. Compare to relational database that the researcher creates for guidelines, Neo4j show same result with relational database for book recommendation by using author's parameter or book type's parameter. Graph database can adjust their own relationship by declare their parameter first. After that, by using cypher, relationship should be created. But, if the graph database has a lot of nodes, each node should manually create, using query, one by one connect the relationship to nodes.

Observations: This paper however uses graph database as NoSQL derivative. Virtually every graph database has to store and query the graph like relational database in general. There has been also an increase of interest in graph that representing social network and web site link structure. The paper has introduced about graph database usage in book recommendation. The paper considers a graph database oriented to represent BibTeX book metadata to create a book recommendation.

According to the author**, Yongen Liang, ShimingWan** School of Big Data and Computer University of Guangdong Baiyun Guangzhou, Guangdong Province, China, Personalized recommendation technology is a new technology which can mine products by using user's information. A recommendation system establishes a user-based model or a project-based model by analysing the user's behavior on the website, such as searching history, browsing history, grading items, and metadata of the project. Personalized recommendation module produces a list of books' recommendations to the readers based on the interest of library readers. Users' interest points are calculated by user borrowing history information through collaborative filtering algorithm. This recommendation system only provides recommendation services to registered users. Usually, if the system does not get the personal information provided by users, users will not be able to obtain the best information service. Relying on a large number of books information, user information, borrowing books information in the library, the user CF and Item CF algorithm are combined to complete the personalized recommendation update.

Observations: The system uses a database and collaborative filtering algorithm along with precision and uses the expert recommendation function to recommend books for new readers and to recommend new books to readers, which is helpful to improve the utilization rate of books and the quality of information service, and realize the unification of personalization and accuracy of University Books recommendation. It also highly focuses on people's feedback. Its user-based collaborative filtering algorithm which consists of a) Finding a collection of books similar to the interest of the target user. b) Find books that the user in this collection likes and that the target user has not heard of, and recommend them to the target user.

According to the author, **Chaloemphon Sirikayon, Panita Thusaranon, Piyalak Pongtawevirat** from College of Innovative Technology and Engineering Dhurakij Pundit University Bangkok, Thailand the proposed system consists of effectiveness and efficiency of book recommendation system as it is a significant issue which could enhance student's performance. This research presents the process of book recommendation by using the collaborative filtering (CF) for university students. The CF technique composes of similarity calculation, prediction and recommendation. In our experiments, matrix factorization technique is also adopted to solve sparsity of rating matrix. Another approach of recommender system is based on data mining techniques such as association rule, clustering, and decision tree. A library book recommendation system based on user profile loaning and apply association rule to create model was proposed by. Association of users, book categories, and book titles are explored to extract rules.

The system states different models have been developed in order to generate book recommendation. Many approaches rely on collaborative filtering (CF) methods based on the main idea that people have similar preferences and interests, so similarities of users or books are calculated. Collaborative Filtering algorithm is based on the main idea that people have similar preferences and interests. he prediction problem is to predict the rating active user Ua will give to an item I_{ua} from the set of all items that U_a has not yet rated. The CF technique composes of 3 steps as follows: 1) users similarity calculation 2) top N nearest neighbors' selection and 3) prediction.

Observation: Collaborative filtering (CF), user based, collaborative filtering, N nearest neighbors' selection. Under Similarity and distance, they have used, Pearson correlation, Cosine similarity, Euclidean distance, Prediction, Matrix Factorization and database. The data used for this study are all 124,406 library records of Dhurakij Pundit University from Jan 1, 2014 to Dec 31, 2017. Based on their similarity scores, most similar k students to active student are then selected and the rating of each book for active student are predicted by using scores of books that k similar neighbors have already taken.

According to the author**, Kitti Puritat** Department of Library and Information Science and **Kannikar Intawong** Department of Public Health Chiang Mai University Chiang Mai, Thailand. In order to make use of the recommendation system in library automation, it was simply applied to the OPAC module (Online Public Access Catalog) which is used as a public interface of users for search and retrieval. A book recommendation was provided by the SVM based on similarities between titles or bibliographic. Moreover, the method is based on the matches/mismatches between NDC categories, and similarities between the outlines in the BOOK Database.

We decided to develop our framework based on OpenBiblio the automated library system open-source framework (http://obiblio.sourceforge.net) because it contains OPAC, circulation, cataloguing, and staff administration which has enough modules for small and medium libraries and most importantly it has the GNU GPL license. We improved the OpenBiblio in terms of the module for Thai languages and it supports new technologies such as the web responsive and PHP 7.0 or higher and the book recommendation system. They have also implemented D.D.C (Dewey decimal classification) is the standard classification method used in various Thai libraries. It is the concept of numbers representing the subject fields as three-digit numbers such as 500 for "Science" and 510 for "mathematic".

Observation: A book recommendation was provided by the SVM based on similarities between titles or bibliographic. Moreover, the method is based on the matches/mismatches between NDC categories, and similarities between the outlines in the BOOK Database. In order to make use of the recommendation system in library automation, it was simply applied to the OPAC module (Online Public Access Catalog). Moreover, the method is based on the matches/mismatches between NDC categories, and similarities between the outlines in the BOOK Database. Also, D.D.C (Dewey decimal classification) is used as the standard classification method.

According to the author**, Madhuri Kommineni, P. Alekhya, T. Mohana Vyshnavi, V. Aparna,** Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, AP, India proposed, A hybrid recommendation system combining the features of collaborative, content based and demographic methods with high accuracy implemented in java. This book recommendation engine introduced is a professional tool to recommend e-user books. The recommender feature is certainly going to be a great Java language web application. This type of web application should prove beneficial for today's highly demanding websites for online purchases. This hybrid recommender system is more accurate and efficient because it combines the characteristics of different recommendation techniques. The recommendation engine of the book should reduce the overhead associated with making the best book choices among the abundance.

The dataset used for this is taken from Kaggle Goodreadsbooks data where you can get the data about books, authors and their titles along with ratings. This system uses good reads data set. This data set is having 7 tables. The dataset is highly accurate and the most recommended dataset to perform various all other actions.

Observation: The system gathers feedback from goodreads-book dataset User-item matrix: User related data and device objects are entered as a list of numerical ratings in a User-Item matrix. Using the similarity methods (e.g., PCC, CPCC, Cosine, Jaccard) we compute similarity matrix for each and every user. The similarity measure ranges from 0, 1. The predictive scores are calculated from the above generated matrix by applying aggregate method-deviation from mean method. The predictive scores might be binary (0/1) or numerical values which depict the score that target user might give for all books based on the ratings given by neighbours and similarities. Sort the scores of all books to get an order.

According the author, **K Swetha, V Mounika,** Department of Computer Science and Engineering, KoneruLakshmaiah Education Foundation, Vaddeswaram, AP, India, proposed a system using CCF using SVD model for news recommendations on Bing and proved to be efficient when compared to other algorithms. The proposed CCF combines both the advantages of the Content-based (CBF) Filtering approach and the features of the Collaborative (CF) Filtering approach by using some of the rich contexts and focusing on long-tail users. This CCF is designed for settings such as the recommendation of the Bing news topic, where a piece of news could be interpreted by rich contexts such as the results of querying.

We also compared some similarity measures and found the best one. The user based collaborative filtering method builds a user-user similarity matrix by using some similar measures to present the similarity between users in order to utilize it for further processing. We can deduct from the results and visualizations that rating the accuracy followed by a normal distribution implies its consistency and efficiency.

Observation: The proposed CCF combines both the advantages of the Content-based (CBF) Filtering approach and the features of the Collaborative (CF) Filtering approach by using some of the rich contexts and focusing on long-tail users. Uses nearest neighbors' algorithms and matrix factorization for social voting and concluded that an affiliations factors play animportant role in increasing the accuracy. Uses nearest neighbors' algorithms and matrix factorization for social voting and concluded that an affiliations factors play an important role in increasing the accuracy.

SUMMARY OF LITERATURE SURVEY

A literature review is an objective, critical summary of published research literature relevant to a topic under consideration for research. The summary is presented here.

Table 1:Summary of Literature Survey

S. No.	Paper	Task	Observation
1.	A Hybrid Model for Book Recommendation - Rohit Darekar, Karan Dayma, Rohan Parabh, Prof. Swapnali Kurhade.	They proposing a novel hybrid book recommender system combining the different recommender algorithms to optimize our recommendations. The system will find out which book the user had bought earlier i.e., the genre or category of the book and then finding out the sub- category if it could have.	Technique: using Collaborative based filtering (CF), Content based filtering (CB) and Demographic based filtering.

2.	I Nyoman Pande Wahyu Dharmawan, Riyanarto Sarno Informatics – The design of disciplinary book recommendation system based.	The system includes great accuracy to find the most recommended books using Platform Login Interface, Information Input of Users, The Main Interface of the Book Recommendation System, The Book Classification and Rating Interface, Book Management Function.	Technique: traditional database system using E-R database system with various input and based on the user's behavior.
3.	Book Recommendation using Neo4j Graph Database in BibTeX Book Metadata.	The system uses a Neo4j provides a cypher query to extract data from graph database. The paper considers a graph database oriented to represent BibTeX book metadata to create a book recommendation. After that, by using cypher, relationship should be created. But, if the graph database has a lot of nodes, each node should manually create, using query, one by one connect the relationship to nodes.	Technique: It uses graph database as NoSQL with graph database and book metadata that is stored in Neo4j. Neo4j itself is a graph database allowing storing data as nodes that connected by arcs.
4.	The Design and Implementation of Books Recommendation System	The system uses a database and collaborative filtering algorithm along with precision and expertise recommendation. It also highly focuses on people's feedback. The system has good accuracy.	Technique: Collaborative based filtering (CF), Database and highly accurate for Expert and New Books recommendation
5	A Collaborative Filtering Based Library Book Recommendation System	The system states many approaches rely on collaborative filtering methods based on the main idea of the people. Based on their similarity scores, most similar students are then selected and the rating of each book for active student scores of books.	Technique: collaborative filtering (CF), user based, collaborative filtering, N nearest neighbors' selection

6	Development of an Open-Source Automated Library System with Book Recommendation System for Small Libraries.	In order to make use of the recommendation system in library automation, it was simply applied to the OPAC module (Online Public Access Catalog). A book recommendation was provided by the SVM based on similarities between titles or bibliographic. A database system and. D.C (Dewey decimal classification) is the standard classification.	Technique: SVM based on similarities, based on the matches/mismatches between NDC categories, and similarities between the outlines in the BOOK Database.
7	Machine Learning Based Efficient Recommendation System for Book Selection using User based Collaborative Filtering Algorithm	The system gathers feedback from goodreads-book dataset User-item matrix: User related data and device objects are entered as a list of numerical ratings in a User-Item matrix. Using the similarity methods (e.g., PCC, CPCC, Cosine, Jaccard) we compute similarity matrix for each and every user. The similarity measure ranges from 0, 1.	Technique: proposed a system using CCF using SVD model for news recommendations on Bing and proved to be efficient when compared to other algorithms.
8	A book recommendation system using Nearest Neighbor Algorithm with matrix factorization.	We also compared some similarity measures and found the best one. The user based collaborative filtering method builds a user-user similarity matrix by using some similar measures to present the similarity between users in order to utilize it for further processing.	Technique: Content-based (CBF) Filtering & Collaborative (CF) Filtering Also, uses nearest neighbors' algorithms and matrix factorization.

RESEARCH GAP

Many systems lack the data i.e., concerning the recommendation system. There is large amount of data required to produce effective and accurate recommendations. One way the problem can be solved is by having many users and accordingly the data will also increase. While collecting the data it becomes difficult to organize and store and use a recommendation for the right user. For a user, which has a new input of recommendation, which database lacks the details to the consumers input doesn’t recommends the accurate outcome.

Hence, to overcome this issue the we need to come with something that whatever the user makes the input the system should be to provide output to it and that too the most precise output with the utmost accuracy. So, this over cover all the recommendation for all types of user whether it can be a casual reader, a school student, college student or a teacher or any other expertise working professional.

CHAPTER 3

THE PROPOSED SYSTEM

Overview of the Recommendation System

There is a large growth in the amount of data available on the internet coupled with the fact that there are a large number of internet users as well. This in turn has led to the problem of exposure to this huge data leading to information overloading. Many IR systems i.e., Informal Retrieval system have tried to solve this problem. They have partially solved the problem but still the large problem lingers on which is known as personalization. Personalization in simple words means tailored for a specific user or a group of users according to his/her interests. Hence for this very purpose of personalization or customization, Recommender systems came into the picture. Recommender system is basically a filtering system customized according to the user's needs.

Recommenders typically use either collaborative filtering or content-based filtering or both. Collaborative sorting methods create a model based on the past experience of a user (items that are liked, rated or purchased previously) and also considering the similar decisions taken by others. Content based filtering method uses a list of discrete, pre-tagged item properties to recommend other items that have similar properties. Present recommendation systems combine in a hybrid system one or more approaches.

The main purpose of the existing system is to provide a recommendation system of books for educational purpose where many of the students lacks in deciding which book is to be referred for a particular topic, which should be the right book for beginners, for teachers, for expertise. It also includes the recommendations of books for the casual reader. The system works with the beautiful soap web scrapping technology to figure out all the required suggestions and rating of a book for the consumer to be recommended. This would also make sure all types of user will get the same or the other outcome with minimum satisfying accuracy. So, this over cover all the recommendation for all types of user whether it can be a casual reader, a school student, college student or a teacher or any other expertise working professional.

EXISTING SYSTEM ARCHITECTURE

The existing system has user profile from where the system can get the user's details and some personalized interests of the user. The system works this way: first the it would ask for the user to input the constraints for which the system will work upon. It would have categories like the science, fiction, non-fiction, crime, kids, comics, school-level books, educational books. It could also have sub-category like for educational books, school level, high school, Jr. college, degree college, researcher's books.

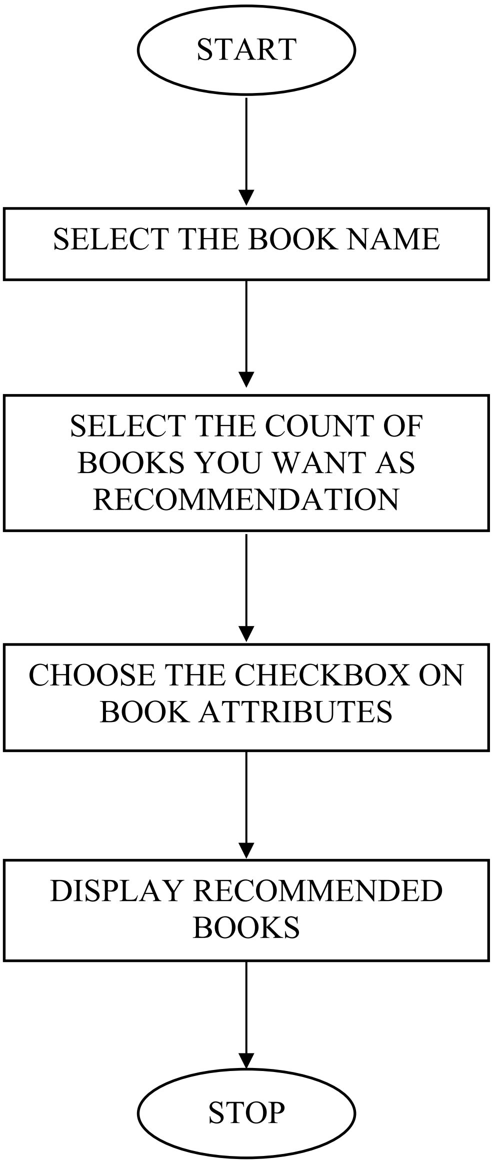

START
SELECT THE BOOK NAME
SELECT THE COUNT OF BOOKS YOU WANT AS RECOMMENDATION
CHOOSE THE CHECKBOX ON BOOK ATTRIBUTES
DISPLAY RECOMMENDED BOOKS
STOP

Figure 1: Flowchart of System Architecture

Different Recommendation Algorithms and Technologies used

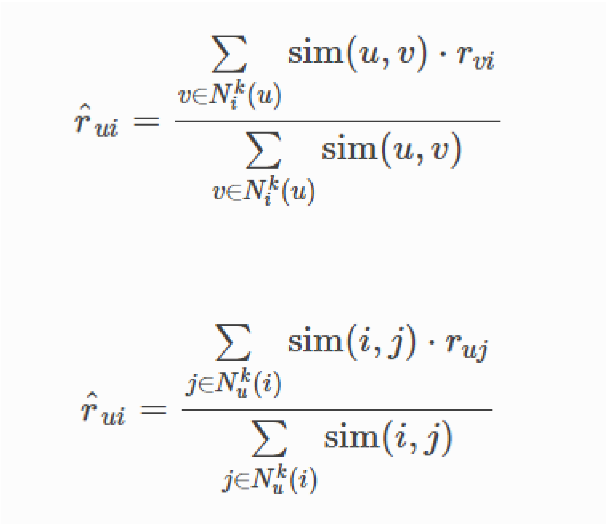
1. Collaborative based filtering (CF)

Collaborative filtering is one of the prominent techniques used in recommender systems. Collaborative filtering as the name itself suggests that it is a method of filtering the interests of a particular user on the basis of collaboration among the various users, variousthoughts, etc. Collaborative filtering involves a large amount of user data and his/her preferences. Collaborative filtering is of two types namely the model based and memory based. Memory based approach requires us to find the similarity measure i.e., we need to find the correlation between two users or a group of users while in Model based approacha model has to be created using various data mining and machine learning algorithms.

2. K-NN Basic

A basic collaborative filtering algorithm.

The prediction r^{ui} is set as:



$$\hat{r}_{ui} = \frac{\sum\limits_{v \in N_i^k(u)} \text{sim}(u, v) \cdot r_{vi}}{\sum\limits_{v \in N_i^k(u)} \text{sim}(u, v)}$$

$$\hat{r}_{ui} = \frac{\sum\limits_{j \in N_u^k(i)} \text{sim}(i, j) \cdot r_{uj}}{\sum\limits_{j \in N_u^k(i)} \text{sim}(i, j)}$$

Figure 2: KNN-Basic Formula

depending on the *user_based* field of the *sim_options* parameter.

Parameters:

- **k (int) –** The (max) number of neighbors to take into account for aggregation (see this note). Default is 40.
- **min_k (int) –** The minimum number of neighbors to take into account for aggregation. If there are not enough neighbors, the prediction is set to the global mean of all ratings. Default is 1.
- **sim_options (dict) –** A dictionary of options for the similarity measure. See Similarity

measure configuration for accepted options.

3. **Content based filtering (CB)**

Content filtering is another prominent technique used in the recommender systems.Content based filtering depends on the user's profile and also the item. As the name suggests this technique is based on the content i.e., recommendations are given on the basisof current content which are the user's likes or dislikes, according to it the items are recommended. A content-based filtering is a learning process it keeps on changing as the user's preference change.

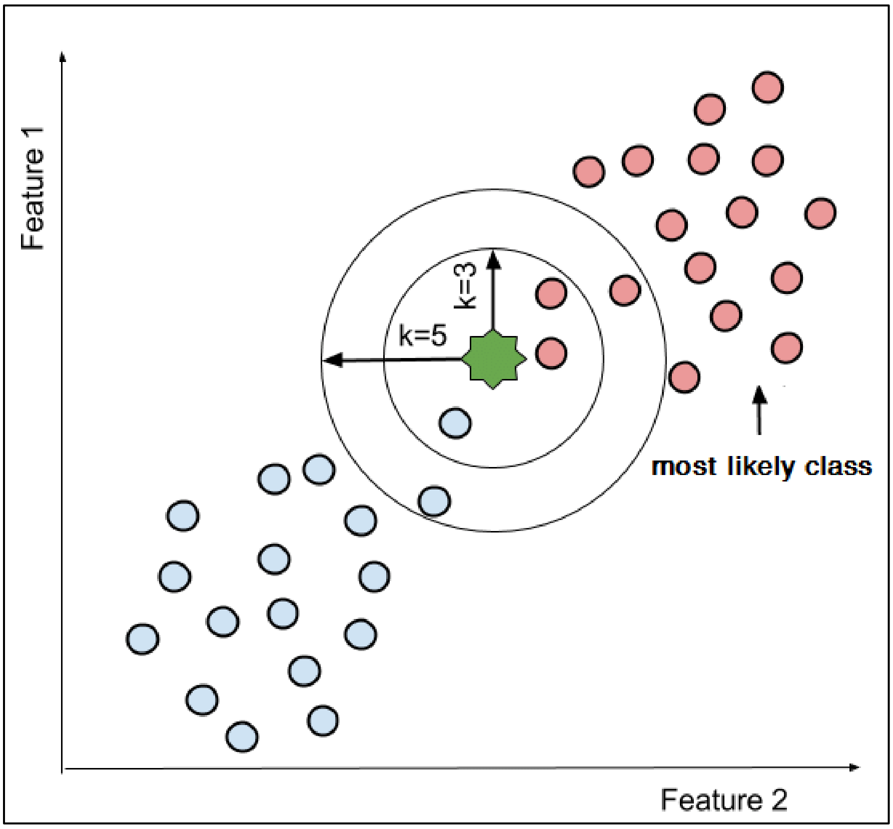


Figure 3:Similarity of Features

4. **Cosine Similarity**

Compute cosine similarity between samples in X and Y.

Cosine similarity, or the cosine kernel, computes similarity as the normalized dot product of X and Y:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure 4:Cosine Similarity Formula

Parameters:

- **X:** {ndarray, sparse matrix} of shape (n_samples_X, n_features)
- Input data.
- **Y:** {ndarray, sparse matrix} of shape (n_samples_Y, n_features), default=None
- Input data. If None, the output will be the pairwise similarities between all samples in X.
- **dense_outputbool, default = True**
- Whether to return dense output even when the input is sparse. If False, the output is sparse if both input arrays are sparse.

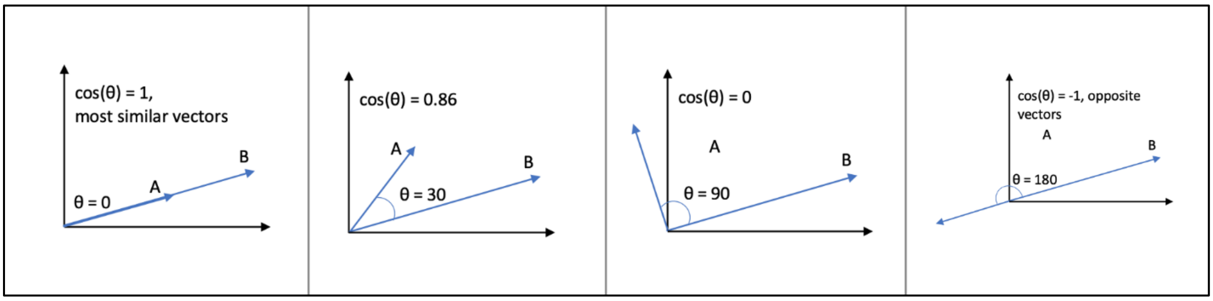


Figure 5:Cosine graph

The numerator of the formula is the dot product of the two vectors and denominator is the product of L2 norm of both the vectors. Dot product of two vectors is the sum of element wise multiplication of the vectors and L2 norm is the square root of sum of squares of elements of a vector.

We can either use inbuilt functions in NumPy library to calculate dot product and L2 norm of the vectors and put it in the formula or directly use the cosine_similarity from sklearn. metrics. Pairwise.

5. Karl Pearson Similarity

We can use the Pearson Similarity algorithm to work out the similarity between two things.

We might then use the computed similarity as part of a recommendation query. For example, to get movie recommendations based on the preferences of users who have given similar ratings to other movies that you've seen.

$$r = \frac{\sum (x - m_x)(y - m_y)}{\sqrt{\sum (x - m_x)^2 \sum (y - m_y)^2}}$$

Figure 6:Pearson Formula

WEB SCRAPPING

Web scraping is the process of collecting structured web data in an automated fashion. It's also called web data extraction. Some of the main use cases of web scraping include price monitoring, price intelligence, news monitoring, lead generation and market research among many other. In general, web data extraction is used by people and businesses who want to make use of the vast amount of publicly available web data to make smarter decisions.

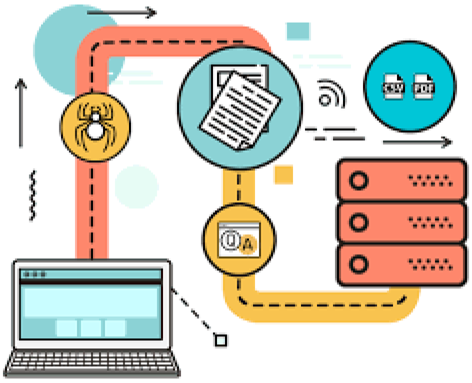


Figure 7:Web Scrapping

If you've ever copy and pasted information from a website, you've performed the same function as any web scraper, only on a microscopic, manual scale. Unlike the mundane, mind-numbing process of manually extracting data, web scraping uses intelligent automation to retrieve hundreds, millions, or even billions of data points from the internet's seemingly endless frontier.

Web Crawler and Web Spider

A web crawler, spider, or search engine bot downloads and indexes content from all over the Internet. The goal of such a bot is to learn what (almost) every webpage on the web is about, so that the information can be retrieved when it's needed. They're called "web crawlers" because crawling is the technical term for automatically accessing a website and obtaining data via a software program. These bots are almost always operated by search engines. By applying a search algorithm to the data collected by web crawlers, search engines can provide relevant links in response to user search queries, generating the list of webpages that show up after a user types a search into Google or Bing (or another search engine).

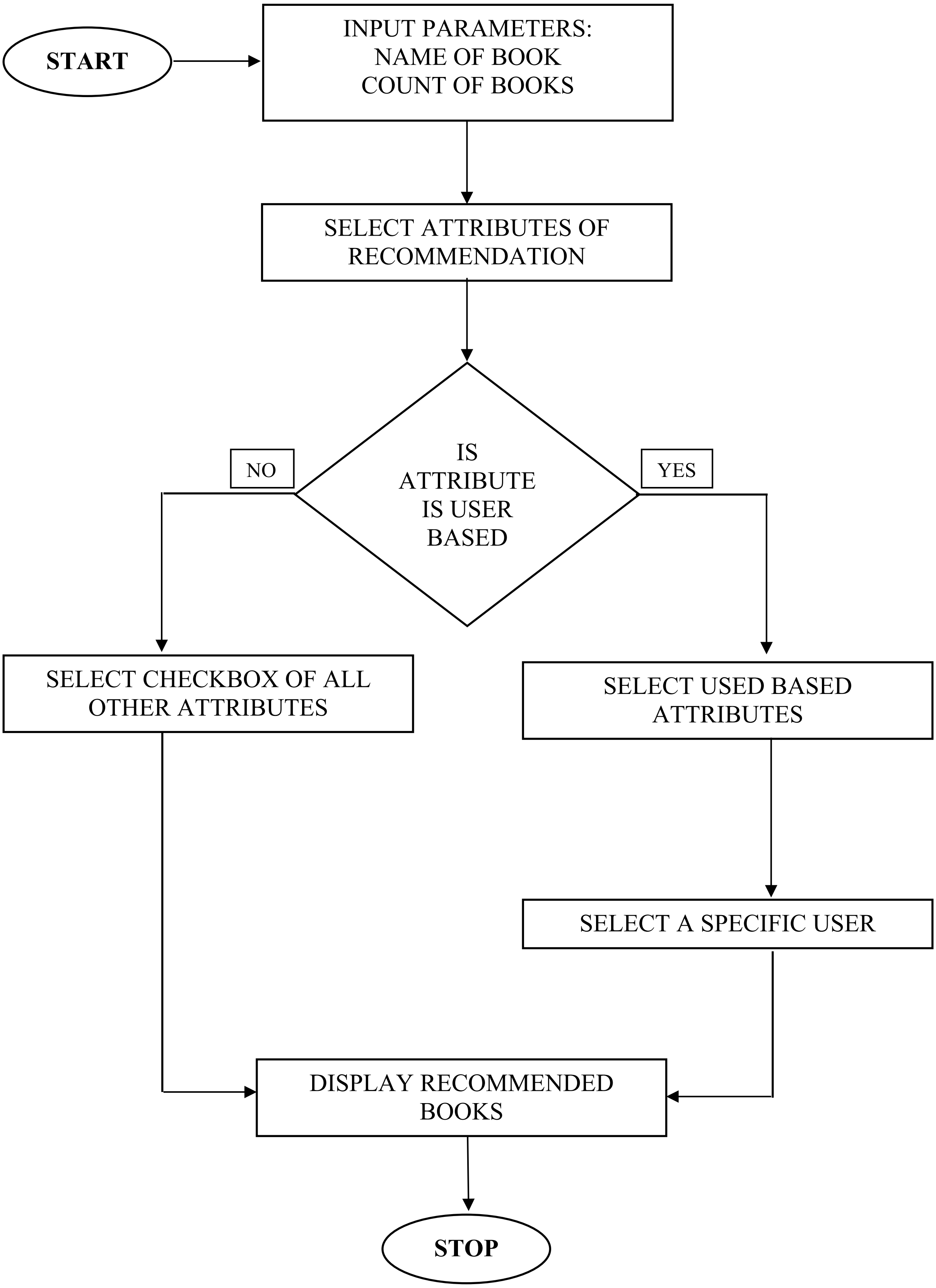
Scrapy is a Python framework for web scraping that provides a complete package for developers without worrying about maintaining code. Beautiful Soup is also widely used for web scraping. It is a Python package for parsing HTML and XML documents and extract data from them. It is available for Python 2.6+ and Python 3.

BeautifulSoup

The incredible amount of data on the Internet is a rich resource for any field of research or personal interest. To effectively harvest that data, you'll need to become skilled at web scraping. The Python libraries requests and Beautiful Soup are powerful tools for the job. If you like to learn with hands-on examples and you have a basic understanding of Python and HTML, then this tutorial is for you. To get information about the elements we want to access, we first of need to inspect the web page using the developer tools.

In this post we will scrape the "content" and "see also" sections from an arbitrary Wikipedia article. To get information about the elements and attributes used for the sections, we can right click on the element to inspect it. This will open the inspector which lets us look at the HTML The created BeautifulSoup object can now be used to find elements in the HTML. When we inspected the website, we saw that every list item in the content section has a class that starts with to section- and we can us Beautiful Soup's find all method to find all list items with that class.

FLOWCHART



START
INPUT PARAMETERS:
NAME OF BOOK
COUNT OF BOOKS
SELECT ATTRIBUTES OF RECOMMENDATION
IS ATTRIBUTE IS USER BASED
NO
YES
SELECT CHECKBOX OF ALL OTHER ATTRIBUTES
SELECT USED BASED ATTRIBUTES
SELECT A SPECIFIC USER
DISPLAY RECOMMENDED BOOKS
STOP

Figure 8:Flowchart of Code

THE DATASET

GoodReads

The dataset used for this is taken from Kaggle GoodReads - Books data where you can get the dataabout books, authors and their titles along with ratings. We commend the books to users based on cosine similarity. The recommendations of a book are being affected by many variables such as category, sub_category, book_id, user_id, best_book_id, tag_id, tag_name, ratings, received_ratings, total_ratings, books_count and similarity_measure. This system uses good reads data set. This data set is having 5 CSV Files.

BOOKS.csv

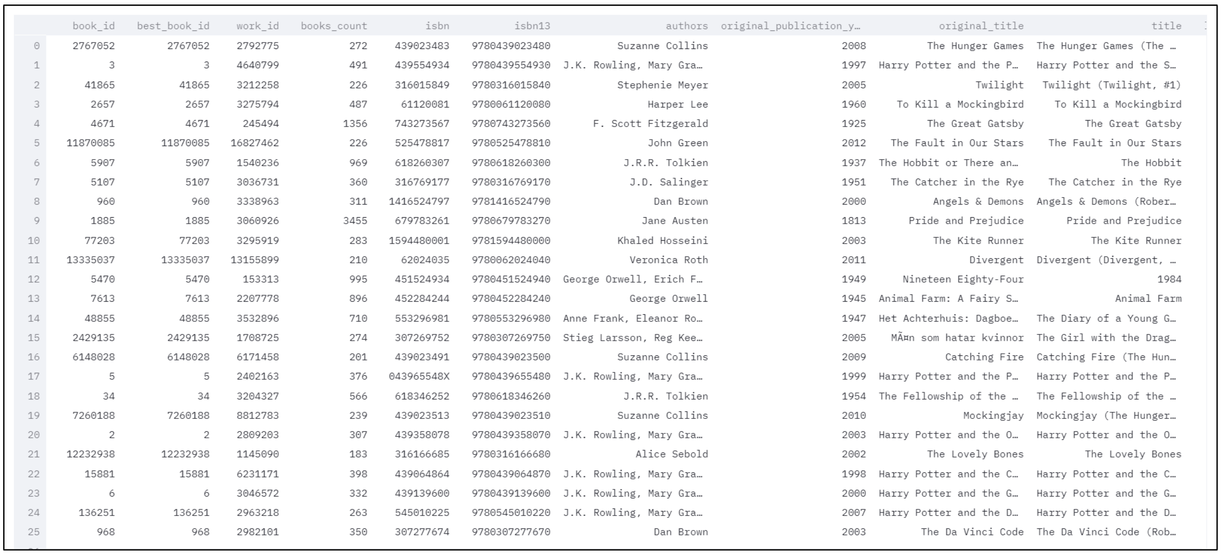
TAGS.csv

BOOK_TAGS.cv

RATINGS.csv

TO_READ.csv

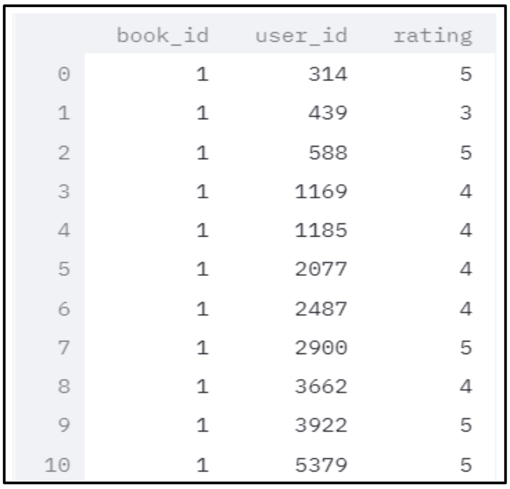
BOOK.csv



	book_id	best_book_id	work_id	books_count	isbn	isbn13	authors	original_publication_y…	original_title	title
0	2767052	2767052	2792775	272	439023483	9780439023480	Suzanne Collins	2008	The Hunger Games	The Hunger Games (The …
1	3	3	4640799	491	439554934	9780439554930	J.K. Rowling, Mary Gra…	1997	Harry Potter and the P…	Harry Potter and the S…
2	41865	41865	3212258	226	316015849	9780316015840	Stephenie Meyer	2005	Twilight	Twilight (Twilight, #1)
3	2657	2657	3275794	487	61120081	9780061120080	Harper Lee	1960	To Kill a Mockingbird	To Kill a Mockingbird
4	4671	4671	245494	1356	743273567	9780743273560	F. Scott Fitzgerald	1925	The Great Gatsby	The Great Gatsby
5	11870085	11870085	16827462	226	525478817	9780525478810	John Green	2012	The Fault in Our Stars	The Fault in Our Stars
6	5907	5907	1540236	969	618260307	9780618260300	J.R.R. Tolkien	1937	The Hobbit or There an…	The Hobbit
7	5107	5107	3036731	360	316769177	9780316769170	J.D. Salinger	1951	The Catcher in the Rye	The Catcher in the Rye
8	960	960	3338963	311	1416524797	9781416524790	Dan Brown	2000	Angels & Demons	Angels & Demons (Rober…
9	1885	1885	3060926	3455	679783261	9780679783270	Jane Austen	1813	Pride and Prejudice	Pride and Prejudice
10	77203	77203	3295919	283	1594480001	9781594480000	Khaled Hosseini	2003	The Kite Runner	The Kite Runner
11	13335037	13335037	13155899	210	62024035	9780062024040	Veronica Roth	2011	Divergent	Divergent (Divergent, …
12	5470	5470	153313	995	451524934	9780451524940	George Orwell, Erich F…	1949	Nineteen Eighty-Four	1984
13	7613	7613	2207778	896	452284244	9780452284240	George Orwell	1945	Animal Farm: A Fairy S…	Animal Farm
14	48855	48855	3532896	710	553296981	9780553296980	Anne Frank, Eleanor Ro…	1947	Het Achterhuis: Dagboe…	The Diary of a Young G…
15	2429135	2429135	1708725	274	307269752	9780307269750	Stieg Larsson, Reg Kee…	2005	MÃ¤n som hatar kvinnor	The Girl with the Drag…
16	6148028	6148028	6171458	201	439023491	9780439023500	Suzanne Collins	2009	Catching Fire	Catching Fire (The Hun…
17	5	5	2402163	376	043965548X	9780439655480	J.K. Rowling, Mary Gra…	1999	Harry Potter and the P…	Harry Potter and the P…
18	34	34	3204327	566	618346252	9780618346260	J.R.R. Tolkien	1954	The Fellowship of the …	The Fellowship of the …
19	7260188	7260188	8812783	239	439023513	9780439023510	Suzanne Collins	2010	Mockingjay	Mockingjay (The Hunger…
20	2	2	2809203	307	439358078	9780439358070	J.K. Rowling, Mary Gra…	2003	Harry Potter and the O…	Harry Potter and the O…
21	12232938	12232938	1145090	183	316166685	9780316166680	Alice Sebold	2002	The Lovely Bones	The Lovely Bones
22	15881	15881	6231171	398	439064864	9780439064870	J.K. Rowling, Mary Gra…	1998	Harry Potter and the C…	Harry Potter and the C…
23	6	6	3046572	332	439139600	9780439139600	J.K. Rowling, Mary Gra…	2000	Harry Potter and the G…	Harry Potter and the G…
24	136251	136251	2963218	263	545010225	9780545010220	J.K. Rowling, Mary Gra…	2007	Harry Potter and the D…	Harry Potter and the D…
25	968	968	2982101	350	307277674	9780307277670	Dan Brown	2003	The Da Vinci Code	The Da Vinci Code (Rob…

Figure 9:GoogReads - Books.csv

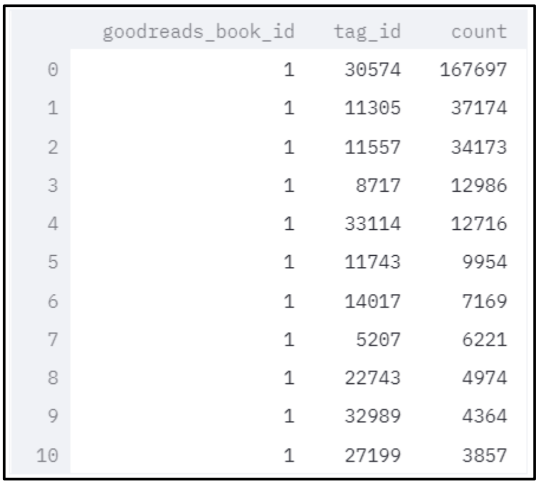
RATINGS.csv



	book_id	user_id	rating
0	1	314	5
1	1	439	3
2	1	588	5
3	1	1169	4
4	1	1185	4
5	1	2077	4
6	1	2487	4
7	1	2900	5
8	1	3662	4
9	1	3922	5
10	1	5379	5

Figure 10:Ratings.csv

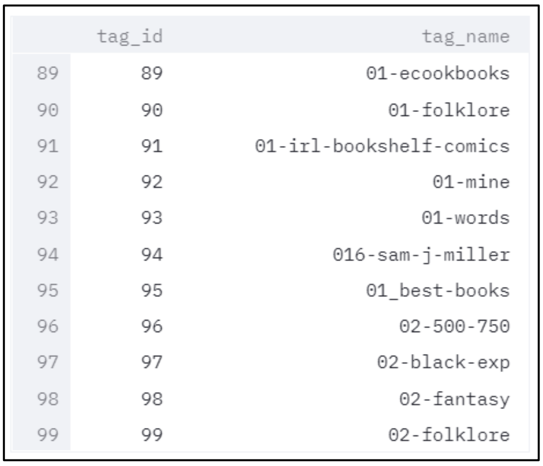
BOOK_TAGS.csv



	goodreads_book_id	tag_id	count
0	1	30574	167697
1	1	11305	37174
2	1	11557	34173
3	1	8717	12986
4	1	33114	12716
5	1	11743	9954
6	1	14017	7169
7	1	5207	6221
8	1	22743	4974
9	1	32989	4364
10	1	27199	3857

Figure 11:Book_tags.csv

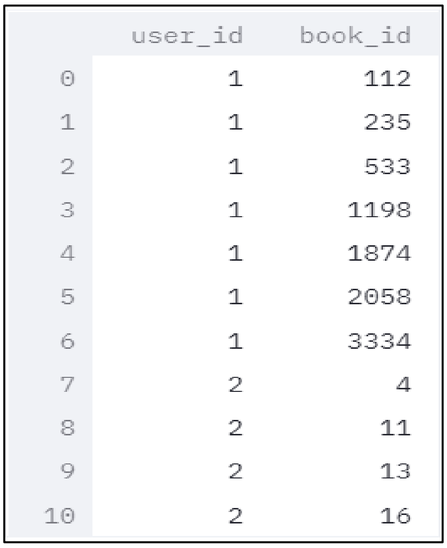
TAGS.csv



	tag_id	tag_name
89	89	01-ecookbooks
90	90	01-folklore
91	91	01-irl-bookshelf-comics
92	92	01-mine
93	93	01-words
94	94	016-sam-j-miller
95	95	01_best-books
96	96	02-500-750
97	97	02-black-exp
98	98	02-fantasy
99	99	02-folklore

Figure 12:Tags.csv

TO_READ.csv



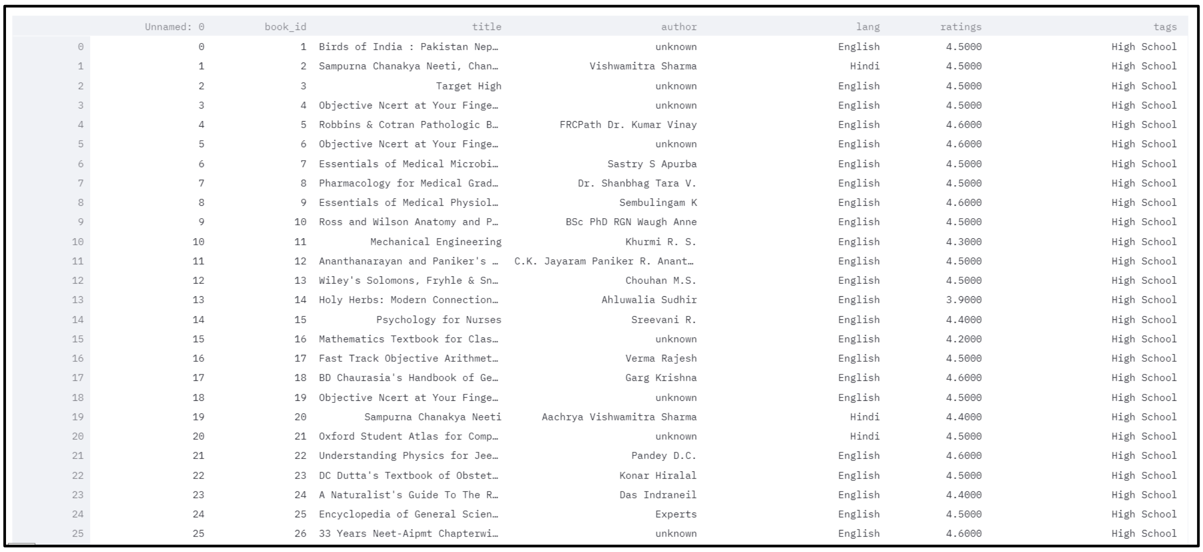
	user_id	book_id
0	1	112
1	1	235
2	1	533
3	1	1198
4	1	1874
5	1	2058
6	1	3334
7	2	4
8	2	11
9	2	13
10	2	16

Figure 13:to_read.csv

Flipkart

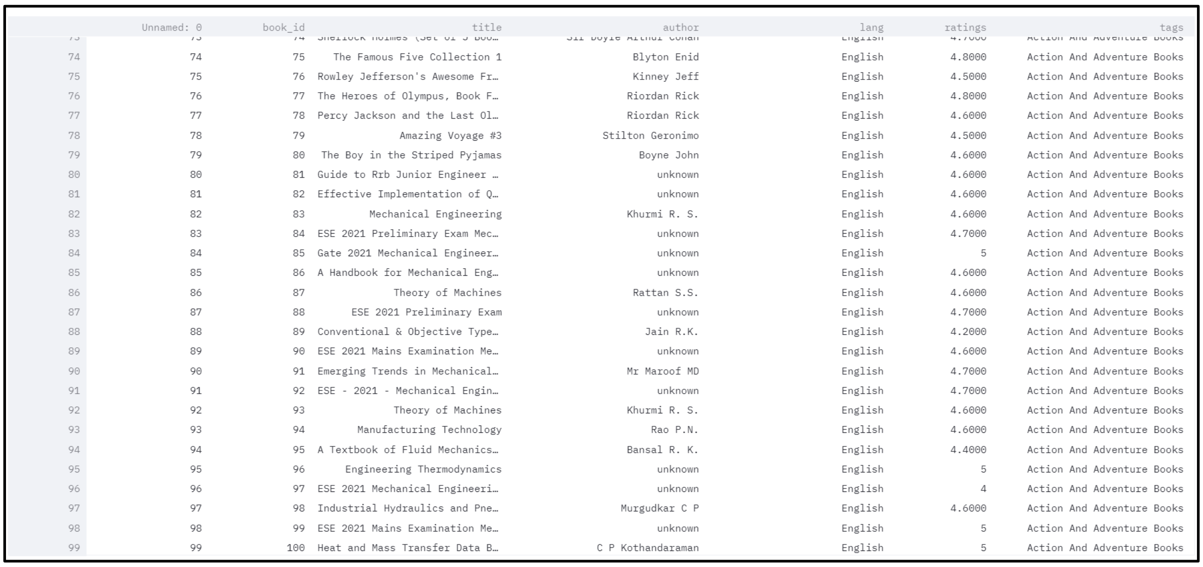
The dataset is extracted from Flipkart - Books section where you can get the data of books, locally sold in India. The most important part is the books, the authors publications are all majorly Indians. Though it is quite obvious that the data needs to be cleaned and made suitable so that mismatching of should be handled properly and mismatching of data and their respective columns. Hence data is extracted from each webpage and cleaned trough Python Programming from each set of 40 records.

BOOK.csv



	Unnamed: 0	book_id	title	author	lang	ratings	tags
0	0	1	Birds of India : Pakistan Nep…	unknown	English	4.5000	High School
1	1	2	Sampurna Chanakya Neeti, Chan…	Vishwamitra Sharma	Hindi	4.5000	High School
2	2	3	Target High	unknown	English	4.5000	High School
3	3	4	Objective Ncert at Your Finge…	unknown	English	4.5000	High School
4	4	5	Robbins & Cotran Pathologic B…	FRCPath Dr. Kumar Vinay	English	4.6000	High School
5	5	6	Objective Ncert at Your Finge…	unknown	English	4.6000	High School
6	6	7	Essentials of Medical Microbi…	Sastry S Apurba	English	4.5000	High School
7	7	8	Pharmacology for Medical Grad…	Dr. Shanbhag Tara V.	English	4.5000	High School
8	8	9	Essentials of Medical Physiol…	Sembulingam K	English	4.6000	High School
9	9	10	Ross and Wilson Anatomy and P…	BSc PhD RGN Waugh Anne	English	4.5000	High School
10	10	11	Mechanical Engineering	Khurmi R. S.	English	4.3000	High School
11	11	12	Ananthanarayan and Paniker's …	C.K. Jayaram Paniker R. Anant…	English	4.5000	High School
12	12	13	Wiley's Solomons, Fryhle & Sn…	Chouhan M.S.	English	4.5000	High School
13	13	14	Holy Herbs: Modern Connection…	Ahluwalia Sudhir	English	3.9000	High School
14	14	15	Psychology for Nurses	Sreevani R.	English	4.4000	High School
15	15	16	Mathematics Textbook for Clas…	unknown	English	4.2000	High School
16	16	17	Fast Track Objective Arithmet…	Verma Rajesh	English	4.5000	High School
17	17	18	BD Chaurasia's Handbook of Ge…	Garg Krishna	English	4.6000	High School
18	18	19	Objective Ncert at Your Finge…	unknown	English	4.5000	High School
19	19	20	Sampurna Chanakya Neeti	Aachrya Vishwamitra Sharma	Hindi	4.4000	High School
20	20	21	Oxford Student Atlas for Comp…	unknown	Hindi	4.5000	High School
21	21	22	Understanding Physics for Jee…	Pandey D.C.	English	4.6000	High School
22	22	23	DC Dutta's Textbook of Obstet…	Konar Hiralal	English	4.5000	High School
23	23	24	A Naturalist's Guide To The R…	Das Indraneil	English	4.4000	High School
24	24	25	Encyclopedia of General Scien…	Experts	English	4.5000	High School
25	25	26	33 Years Neet-Aipmt Chapterwi…	unknown	English	4.6000	High School

Figure 14:Flipkart - book.csv (a)



	Unnamed: 0	book_id	title	author	lang	ratings	tags
73	73	74	Sherlock Holmes (Set of 5 Boo…	Sir Doyle Arthur Conan	English	4.7000	Action And Adventure Books
74	74	75	The Famous Five Collection 1	Blyton Enid	English	4.8000	Action And Adventure Books
75	75	76	Rowley Jefferson's Awesome Fr…	Kinney Jeff	English	4.5000	Action And Adventure Books
76	76	77	The Heroes of Olympus, Book F…	Riordan Rick	English	4.8000	Action And Adventure Books
77	77	78	Percy Jackson and the Last Ol…	Riordan Rick	English	4.6000	Action And Adventure Books
78	78	79	Amazing Voyage #3	Stilton Geronimo	English	4.5000	Action And Adventure Books
79	79	80	The Boy in the Striped Pyjamas	Boyne John	English	4.6000	Action And Adventure Books
80	80	81	Guide to Rrb Junior Engineer …	unknown	English	4.6000	Action And Adventure Books
81	81	82	Effective Implementation of Q…	unknown	English	4.6000	Action And Adventure Books
82	82	83	Mechanical Engineering	Khurmi R. S.	English	4.6000	Action And Adventure Books
83	83	84	ESE 2021 Preliminary Exam Mec…	unknown	English	4.7000	Action And Adventure Books
84	84	85	Gate 2021 Mechanical Engineer…	unknown	English	5	Action And Adventure Books
85	85	86	A Handbook for Mechanical Eng…	unknown	English	4.6000	Action And Adventure Books
86	86	87	Theory of Machines	Rattan S.S.	English	4.6000	Action And Adventure Books
87	87	88	ESE 2021 Preliminary Exam	unknown	English	4.7000	Action And Adventure Books
88	88	89	Conventional & Objective Type…	Jain R.K.	English	4.2000	Action And Adventure Books
89	89	90	ESE 2021 Mains Examination Me…	unknown	English	4.6000	Action And Adventure Books
90	90	91	Emerging Trends in Mechanical…	Mr Maroof MD	English	4.7000	Action And Adventure Books
91	91	92	ESE - 2021 - Mechanical Engin…	unknown	English	4.7000	Action And Adventure Books
92	92	93	Theory of Machines	Khurmi R. S.	English	4.6000	Action And Adventure Books
93	93	94	Manufacturing Technology	Rao P.N.	English	4.6000	Action And Adventure Books
94	94	95	A Textbook of Fluid Mechanics…	Bansal R. K.	English	4.4000	Action And Adventure Books
95	95	96	Engineering Thermodynamics	unknown	English	5	Action And Adventure Books
96	96	97	ESE 2021 Mechanical Engineeri…	unknown	English	4	Action And Adventure Books
97	97	98	Industrial Hydraulics and Pne…	Murgudkar C P	English	4.6000	Action And Adventure Books
98	98	99	ESE 2021 Mains Examination Me…	unknown	English	5	Action And Adventure Books
99	99	100	Heat and Mass Transfer Data B…	C P Kothandaraman	English	5	Action And Adventure Books

Figure 15:Flipkart - book.csv (b)

UI DESIGN

The User interface of the system is such design that every user can easily understand the workflow and mechanism of the system. The system is designed using Python UI building library called StreamLit.

StreamLit

The StreamLit has beautiful, enhanced, simple and pre-styled with CSS and JavaScript in it which allows users to create stunning and beautiful UI for ML and projects. Streamlit is company of tinkerers, engineers, and scientists. We believe that machine learning engineers deserve blazingly fast, fun, and interactive tools. Together, we are building the world's most beautiful tool for machine learning engineers. Streamlit makes it easy for you to visualize, mutate, and share data. The API reference is organized by activity type, like displaying data or optimizing performance. Each section includes methods associated with the activity type, including examples.

Figure 16:streamLit Logo

Streamlit's open-source app framework is the easiest way for data scientists and machine learning engineers to create beautiful, performant apps in only a few hours! All in pure Python. All for free. To install the package ***`pip install streamlit.`*** A few of the advantages of using Streamlit tools like Dash and Flask:

- It embraces Python scripting; No HTML knowledge is needed!
- Less code is needed to create a beautiful application
- No callbacks are needed since widgets are treated as variables
- Data caching simplifies and speeds up computation pipelines.

CHAPTER 4

EXPERIMENTS PERFORMED

In this chapter what are the experiment performed what was the logic, tricks, methods and the way implementation has performed is been explained in a brief manner further below. The system comprises following techniques majorly comprising of four different sections using these technologies:

- TFID vectorization
- SkLearn Metrics
- KNN-Basics
- Beautiful soup

Code:

main.py:

```
### ********************** BOOKS - GOODREADS.CSV ********************
### **PRE-REQUSITES**
import time
import streamlit as st
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import surprise
import plotly_express as px
import warnings
import base64
from PIL import Image
from wordcloud import WordCloud, STOPWORDS
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import linear_kernel
from surprise import KNNBasic, accuracy
from surprise.model_selection import train_test_split
warnings.filterwarnings('ignore')

# -----------------------------------------------------------------------
# BACKEND CODE - READING CSV FILES
# Importing all data
books = pd.read_csv('DATA\\books.csv', encoding="ISO-8859-1")
ratings = pd.read_csv('DATA\\ratings.csv', encoding="ISO-8859-1")
book_tags = pd.read_csv('DATA\\book_tags.csv', encoding="ISO-8859-1")
tags = pd.read_csv('DATA\\tags.csv')
to_read = pd.read_csv('DATA\\to_read.csv')

# some other variables
tags_join_DF = pd.merge(book_tags, tags, left_on='tag_id',
right_on='tag_id', how='inner')
books_merge_ratings = pd.merge(books, ratings)
```

```
books_df = pd.DataFrame(books)
userRatings = books_merge_ratings.pivot_table(index=['user_id'],
columns=['title'], values='rating')
userRatings = userRatings.dropna(thresh=10, axis=1).fillna(0)
all_book_name = userRatings.columns
books_with_tags = pd.merge(books, tags_join_DF, left_on='book_id',
right_on='goodreads_book_id', how='inner')
#-----------------------------------------------------------------------
               # Self defined methods - EDA SECTION

def cloud_author(original_title_string, max_word, max_font, random):
    stop_words = set(STOPWORDS)
    original_title_string = " ".join(books['authors'])
    wc = WordCloud(background_color="white", colormap="hot",
max_words=max_word,
                   stopwords=stop_words, max_font_size=max_font,
random_state=random).generate(original_title_string)
    plt.figure(figsize=(100, 400))
    fig, axes = plt.subplots(1, 2, gridspec_kw={'width_ratios': [8,
1]})
    axes[0].imshow(wc, interpolation="bilinear")
    for ax in axes:
        ax.set_axis_off()
    st.pyplot(fig)

def cloud_title(authors_string, max_word, max_font, random):
    stop_words = set(STOPWORDS)
    authors_string = " ".join(books['title'])
    wc = WordCloud(background_color="white", colormap="hot",
max_words=max_word,
                   stopwords=stop_words, max_font_size=max_font,
random_state=random).generate(authors_string)
    plt.figure(figsize=(100, 400))
    fig, axes = plt.subplots(1, 2, gridspec_kw={'width_ratios': [8,
1]})
    axes[0].imshow(wc, interpolation="bilinear")
    for ax in axes:
        ax.set_axis_off()
    st.pyplot(fig)

#************************************
i = 100
rows_count = []
while (i >= 1):
    rows_count.append(i)
    i = i - 1
#************************************

# ---------------------------------------------------------------------
                              #UI Codes - SIDEBAR
pages = ["Get Recommendation", "Exploratory Data Analysis", "The Data"]

st.sidebar.header("RECOMMENDATION SYSTEM")
st.sidebar.write("Based on Book Author")
st.sidebar.write("Based on Book Tags")
st.sidebar.write("User's Preference")
st.sidebar.write("Item's Preference")
st.sidebar.header("GET INSIGHTS FROM DATA")
st.sidebar.write("check for null values")
st.sidebar.header("ABOUT THE DATA")
st.sidebar.write("books.csv")
st.sidebar.write("ratings.csv")
st.sidebar.write("book_tags.csv")
st.sidebar.write("tags")
st.sidebar.write("to_read.csv")

section1 = st.sidebar.selectbox('', pages)

#-----------------------------------------------------------------------
                    # UI Codes - RECOMMENDATION SECTION
#common section above
if section1 == "Get Recommendation":
    st.markdown("# RECOMMENDATION SYSTEM")

    #for gif image
    file_ = open("confused_lines.gif", "rb")
    contents = file_.read()
    data_url = base64.b64encode(contents).decode("utf-8")
    file_.close()
    st.markdown(f'<img src="data:image/gif;base64,{data_url}" alt="gif"
width = "750">',unsafe_allow_html=True)

#                ---- This is for Book input section ----
    st.subheader("Provide us Book details:")
    book_name_selectbox = books_df['title'].tolist()
    option = st.selectbox(label='Select one:',
options=book_name_selectbox)

    i=20
    list_of_recom = []
    while (i >= 1):
        list_of_recom.append(i)
        i = i-1
    innercol1, innercol2, innercol3 = st.beta_columns(3)
    innercol1.write("You've selected:")
    innercol2.info(option)
    no_of_recom = innercol3.selectbox('No. of Recommendations:',
list_of_recom)

#                    ---- AUTHOR/TAGS/BOTH/ITEM UI ----

    st.subheader("Get Recommendation based on Book attributes:")
    st.write("By which way would you should be recommended:")
    col1, col2, col3, col4 = st.beta_columns(4)
    checkbox1 = col1.checkbox("BOOK AUTHOR")
    checkbox2 = col2.checkbox("BOOK TAGS")
    checkbox3 = col3.checkbox("BOTH")
    checkbox4 = col4.checkbox("ITEM PREF")
#                          --- USER ---
    checkbox5 = st.checkbox("USER'S PREFERENCE")
    if checkbox5 == True:
        my_expander = st.beta_expander("Select User:")
        all_user_list = ratings['user_id'].head(100).tolist()
        selected_user_id = my_expander.selectbox("Select a user: ",
options=all_user_list)
```

```
        rating_count = ratings[ratings['user_id'] == selected_user_id]
        my_expander.write(rating_count)
    submit_btn2 = st.button("SUBMIT", key=1)
#----------------------------------------------------------------------
                  # Self defined methods - USER PREFERENCE
#                        -- PEARSON SIMILARITY --
    item_similarity_pearson = userRatings.corr(method='pearson')
    #50 seconds to run this training set

    ratings = ratings[['user_id','book_id','rating']]
    ratings = ratings.iloc[:20000,:]
    reader = surprise.Reader(rating_scale=(1,5))
    dataset1 = surprise.Dataset.load_from_df(ratings, reader)

    #                                       --------  KNN BASIC MODEL
---------
    from surprise import KNNBasic, accuracy
    from surprise.model_selection import train_test_split
    train1,test1 = train_test_split(dataset1,test_size=0.2)
#----------------------------------------------------------------------
#  ******* Radio click recommendation code logic goes here *******
#                  --- AUTHOR / BOOK TAG / BOTH ---
                        -- Author Based --
    if submit_btn2 == True and checkbox1 == True:
        progress_bar = st.progress(0)
        for i in range(50):
            time.sleep(1)
            progress_bar.progress(i + 1)
        st.info("These are the Recommendations based on Author's
Preference")

        tf = TfidfVectorizer(analyzer='word', ngram_range=(1, 2),
min_df=0, stop_words='english')
        tfidf_matrix = tf.fit_transform(books['authors'])
        cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
        titles = books['title']
        indices = pd.Series(books.index, index=books['title'])

        def authors_recommendation(title):
            idx = indices[title]
            sim_scores = list(enumerate(cosine_sim[idx]))
            sim_scores = sorted(sim_scores, key=lambda x: x[1],
reverse=True)
            sim_scores = sim_scores[1:21]
            book_indices = [i[0] for i in sim_scores]
            return titles.iloc[book_indices]

        result_df = authors_recommendation(option).head(no_of_recom)
        st.table(result_df)
        st.success('I think these are the right book(s) for you!
:smile:')

#                       -- Books tags Based --
    if submit_btn2 == True and checkbox2 == True:
        progress_bar2 = st.progress(0)
        for i in range(50):
            time.sleep(1)
            progress_bar2.progress(i + 1)
        st.info("These are the Recommendations based on Book Tags's
Preference")

        books_with_tags = pd.merge(books, tags_join_DF,
left_on='book_id', right_on='goodreads_book_id', how='inner')
        tf1 = TfidfVectorizer(analyzer='word', ngram_range=(1, 2),
min_df=0, stop_words='english')
        tfidf_matrix1 =
tf1.fit_transform(books_with_tags['tag_name'].head(10000))
        cosine_sim1 = linear_kernel(tfidf_matrix1, tfidf_matrix1)
        titles1 = books['title']
        indices1 = pd.Series(books.index, index=books['title'])

        def tags_recommendation(title):
            idx = indices1[title]
            sim_scores = list(enumerate(cosine_sim1[idx]))
            sim_scores = sorted(sim_scores, key=lambda x: x[1],
reverse=True)
            sim_scores = sim_scores[1:21]
            book_indices = [i[0] for i in sim_scores]
            return titles1.iloc[book_indices]

        result_df = tags_recommendation(option).head(no_of_recom)
        st.table(result_df)
        st.success('I think these are the right book(s) for you!
:smile:')

#                             -- Both --
    if submit_btn2 == True and checkbox3 == True:
        progress_bar3 = st.progress(0)
        for i in range(50):
            time.sleep(1)
            progress_bar3.progress(i + 1)
        st.info("BOTH")
        temp_df =
books_with_tags.groupby('book_id')['tag_name'].apply('
'.join).reset_index()
        books = pd.merge(books, temp_df, left_on='book_id',
right_on='book_id', how='inner')
        books['corpus'] = (pd.Series(
            books[['authors', 'tag_name']]
                .fillna('')
                .values
                .tolist())
                           .str.join(' '))
        tf_corpus = TfidfVectorizer(analyzer='word', ngram_range=(1,
2), min_df=0, stop_words='english')
        tfidf_matrix_corpus = tf_corpus.fit_transform(books['corpus'])
        cosine_sim_corpus = linear_kernel(tfidf_matrix_corpus,
tfidf_matrix_corpus)
        titles = books['title']
        indices1 = pd.Series(books.index, index=books['title'])

        def corpus_recommendation(title):
            idx = indices1[title]
            sim_scores = list(enumerate(cosine_sim_corpus[idx]))
            sim_scores = sorted(sim_scores, key=lambda x: x[1],
reverse=True)
```

```
            sim_scores = sim_scores[1:21]
            book_indices = [i[0] for i in sim_scores]
            return titles.iloc[book_indices]

        result_df = corpus_recommendation(option).head(no_of_recom)
        st.table(result_df)
        st.success('I think these are the right book(s) for you!
:smile:')

#                          -- ITEM --
        if submit_btn2 == True and checkbox4 == True:
            progress_bar4 = st.progress(0)
            for i in range(100):

                st.info("These are the Recommendations based on your
Ratings")

                def get_similar_book_pearson_itemtoitem(bookname):
                    similar_score = item_similarity_pearson[bookname] *
(2.5)
                    similar_score =
similar_score.sort_values(ascending=False)
                    return similar_score

                item_model = surprise.KNNBasic(k=40,
sim_options={'name': 'pearson', 'user_based': False})
                item_model.fit(train1)
                preds = item_model.test(test1)
                accuracy.rmse(preds, verbose=True)

                recommended_pearson_item =
get_similar_book_pearson_itemtoitem(book_name_selectbox).index
                result_df = pd.DataFrame(data=recommended_pearson_item)
                result_df.iloc[1:].head(10)
                result_list = []
                for ind in recommended_pearson_item[0:2]:
                    result_list.append(ind)
                result_df = pd.DataFrame(data=result_list)
                st.table(result_df)
                progress_bar4.progress(i + 1)
            st.success('I think these are the right book(s) for you!
:smile:')
#                               -- USER --

    if submit_btn2 == True and checkbox5 == True:
    #       setting up data of specific user's ratings given to books
            progress_bar4 = st.progress(0)
            for i in range(10):
                time.sleep(1)
                progress_bar4.progress(i + 1)

            my_df = pd.merge(books,rating_count, left_index=True,
right_index=True)
            my_df = my_df.loc[:,
my_df.columns.intersection(['title','rating'])]
            user1_tuple = my_df.to_records(index=False)
            user1_list = list(user1_tuple)
            st.info("These are the Recommendations based on your
Ratings")
            #st.write("User", selected_user_id, "has rated",
user1_list)
            user1_list = zip(all_book_name, userRatings.iloc[0, :])

    #       USER BASED PEARSON SIMILARITY
            item_similarity_pearson =
userRatings.corr(method='pearson')
            # 50 seconds to run this training set

            ratings = ratings[['user_id', 'book_id', 'rating']]
            ratings = ratings.iloc[:20000, :]
            reader = surprise.Reader(rating_scale=(1, 5))
            dataset1 = surprise.Dataset.load_from_df(ratings, reader)

            def get_similar_book_pearson_user(book_name, user_rating):
                      similar_score =
item_similarity_pearson[book_name] * (user_rating-2.5)
                      similar_score =
similar_score.sort_values(ascending=False)
                      return similar_score

            user_model = surprise.KNNBasic(k=40,sim_options={'name':
'pearson','user_based': True})
            user_model.fit(train1)
            preds = user_model.test(test1)
            accuracy.rmse(preds,verbose=True)

            similar_book = pd.DataFrame()
            for bk_name, rating in user1_list:
                similar_book =
similar_book.append(get_similar_book_pearson_user(bk_name, rating),
ignore_index=True)
            recommended_pearson =
similar_book.sum().sort_values(ascending=False).index
            result_list = []
            for ind in recommended_pearson[:10]:
                result_list.append(ind)
            result_df = pd.DataFrame(data=result_list)
            st.table(result_df)
            st.success('I think these are the right book(s) for you!
:smile:')
#-----------------------------------------------------------------------
                            # UI Codes - EDA
# This is for EDA section
if section1 == "Exploratory Data Analysis":
    st.markdown("# GET INSIGHTS FROM DATA")

    # ** check for null values **
    st.subheader("check for null values")
    books_merge_ratings=pd.merge(books, ratings)
    fig, ax = plt.subplots()
    plt.figure(figsize=(12,8))
    sns.heatmap(books_merge_ratings.isnull(), ax=ax, cbar = True)
    st.write(fig)

    # ** which rating is highest from 1- 5 **
    # distribution of average ratings of all the 10000 books
    st.subheader("Which Ratings are more on books")
    plt.title("Distribution of Average Ratings")
    histogram = books["average_rating"]
    st.line_chart(data=histogram)

    # ** which author has more books **
    st.subheader("Which Author has more Books")
    top_author_counts = books['authors'].value_counts().reset_index()
    top_author_counts.columns = ['value', 'count']
    top_author_counts['value'] = top_author_counts['value']
    top_author_counts = top_author_counts.sort_values('count')
    fig = px.bar(top_author_counts.tail(10), x="count", y="value",
orientation='h', color='value',width=800, height=600)
    st.write(fig)

    # ** Which year has maximum number of books published **
    st.subheader("Which year has maximum number of books published")
    years =
books['original_publication_year'].value_counts().reset_index()
    years.columns = ['year', 'count']
    years['year'] = years['year']
    years = years.sort_values('count')
    fig = px.bar(years.tail(50), x="count", y="year", orientation='h',
color='count',width=800, height=600)
    st.write(fig)

    #                  ** Count of Book's Langauge **
    st.subheader("Count of Book's Langauges")
    lang = books['language_code'].value_counts().reset_index()
    lang.columns = ['value', 'count']
    lang['value'] = lang['value']
    lang = lang.sort_values('count')
    fig = px.bar(lang.tail(10), x="count", y="value", orientation='h',
color='count',width=800, height=600)
    st.write(fig)

    # ** which books has highest no of average rating **
    st.subheader("Which books has the highest no of Average Ratings")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=7)
    books_filter = pd.DataFrame(books, columns=['book_id', 'authors',
'original_title', 'average_rating'])
    books_filter = books_filter.sort_values('average_rating',
ascending=False)
    st.write(books_filter.head(20))
    #books_filter_chart =
books_filter.drop(columns={'book_id','average_rating'})
    #st.bar_chart(books_filter_chart.head(20))

    #                  ** WordCloud for Book Title **
    st.markdown("## **Word Cloud - Book Author**")
    authors_string = " ".join(books['authors'])
    my_expander = st.beta_expander("Customize here:")
    max_word1 = my_expander.slider("Set words", 200, 1000, 500, key=1)
    max_font1 = my_expander.slider("Set Font Size", 50, 350, 60, key=2)
    random1 = my_expander.slider("Set Random State", 30, 100, 42,
key=3)
    st.write(cloud_author(authors_string, max_word1, max_font1,
random1))

    # ** WordCloud for Book Title**
    st.markdown("## **Word Cloud - Book Title**")
    my_expander1 = st.beta_expander("Customize here:")
    original_title_string = " ".join(books['title'])
    max_word2 = my_expander1.slider("Set words", 200, 1000, 500)
    max_font2 = my_expander1.slider("Set Max Font Size", 50, 350, 60)
    random2 = my_expander1.slider("Set Random State", 30, 100, 42)

st.write(cloud_title(original_title_string,max_word2,max_font2,random2)
)

    #                  ** Total number of users **
    st.header("Total Number of Users")
    total_users = ratings['user_id'].unique()[-1]
    st.markdown(total_users)

    # ** A specific user has what rating to which book**"""
    # input user_id and get the count of books he rated
    st.markdown("## **Count of books a user rated**")
    all_user_list = ratings['user_id'].head(100).tolist()
    selected_user_id = st.selectbox("Select a user:
",options=all_user_list)
    my_expander2 = st.beta_expander("Select counts:")
    i=20
    user_counts_list = []
    while(i>=1):
        user_counts_list.append(i)
        i=i-1
    user_counts = my_expander2.selectbox("drop down:",
options=user_counts_list)
    rating_count = ratings[ratings['user_id'] ==
selected_user_id].head(user_counts)
    st.write("User",selected_user_id ,"has rated", rating_count)

    # ** check CORRELATION **
    st.subheader("check for CORRELATION:")
    books_merge_ratings = pd.merge(books, ratings)
    fig, ax = plt.subplots()
    plt.figure(figsize=(12, 8))
    sns.heatmap(books_merge_ratings.corr(), ax=ax, cbar=True)
    st.write(fig)

    #------------------------------------------------------------------
    #                         UI Codes - BOOKS DATASET
# This is for books data section
if section1 == "The Data":
    st.markdown("# ABOUT THE DATA")
    st.write("This data is extracted from GoodReads")
    image3 = Image.open('banner_pic.jpg')
    st.image(image3)

    st.header("Get Recommendation based on Book attributes like:")
    st.write("The Data set includes:")

    #BOOK SECTION
    st.markdown("## **BOOKS.csv**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=1)
    st.dataframe(books.head(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(books.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(books.isnull().sum())

    # RATINGS SECTION
    st.markdown("## **RATINGS.csv**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=2)
    st.dataframe(ratings.head(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(ratings.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(ratings.isnull().sum())

    # BOOK_TAGS SECTION
    st.markdown("## **BOOK_TAGS.csv**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=3)
    st.dataframe(book_tags.head(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(book_tags.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(book_tags.isnull().sum())

    # TAGS SECTION
    st.markdown("## **TAGS.csv**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=4)
    st.dataframe(tags.head(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(tags.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(tags.isnull().sum())

    # TO_READ SECTION
    st.markdown("## **TO_READ.csv**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=5)
    st.dataframe(to_read.head(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(to_read.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(to_read.isnull().sum())

    # TAGS_JOIN_DF SECTION
    st.markdown("## **TAGS with BOOKS**")
    selected_rows = st.selectbox("Select no. of rows:",
options=rows_count, key=6)
    st.dataframe(tags_join_DF.tail(selected_rows))
    st.subheader("The Rows and Columns it contain:")
    st.code(tags_join_DF.shape)
    st.subheader("How many empty cells does it contain:")
    st.text(tags_join_DF.isnull().sum())
```

```
#-------------------------------------------------------------------
```

Output:

Which rating is highest from 1to 5

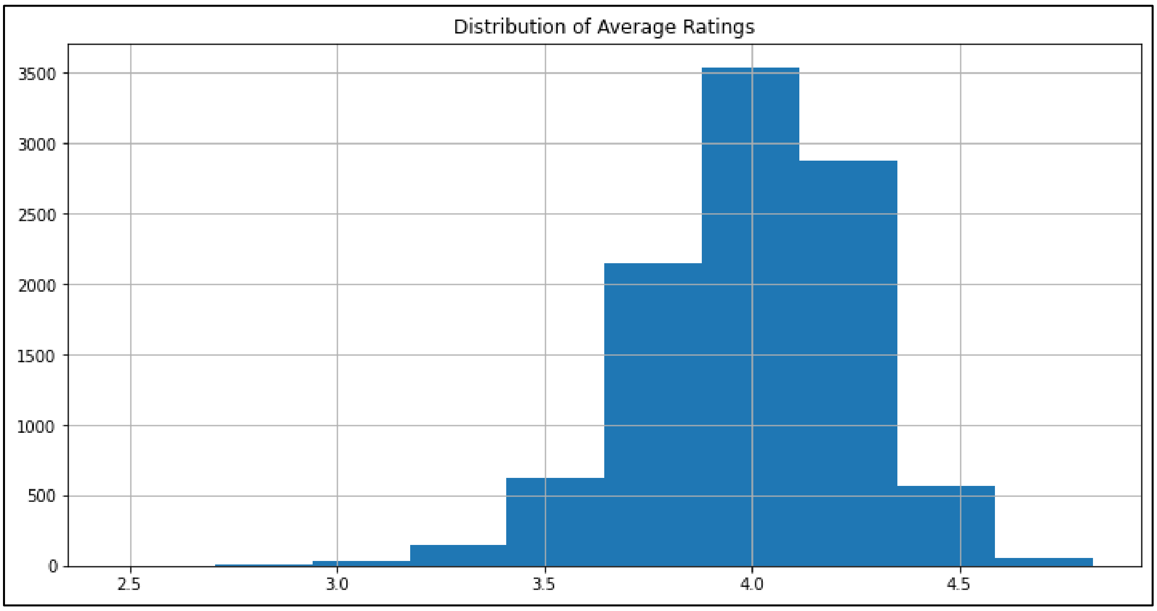


Figure 17:EDA-Graph-1

Which author has more books

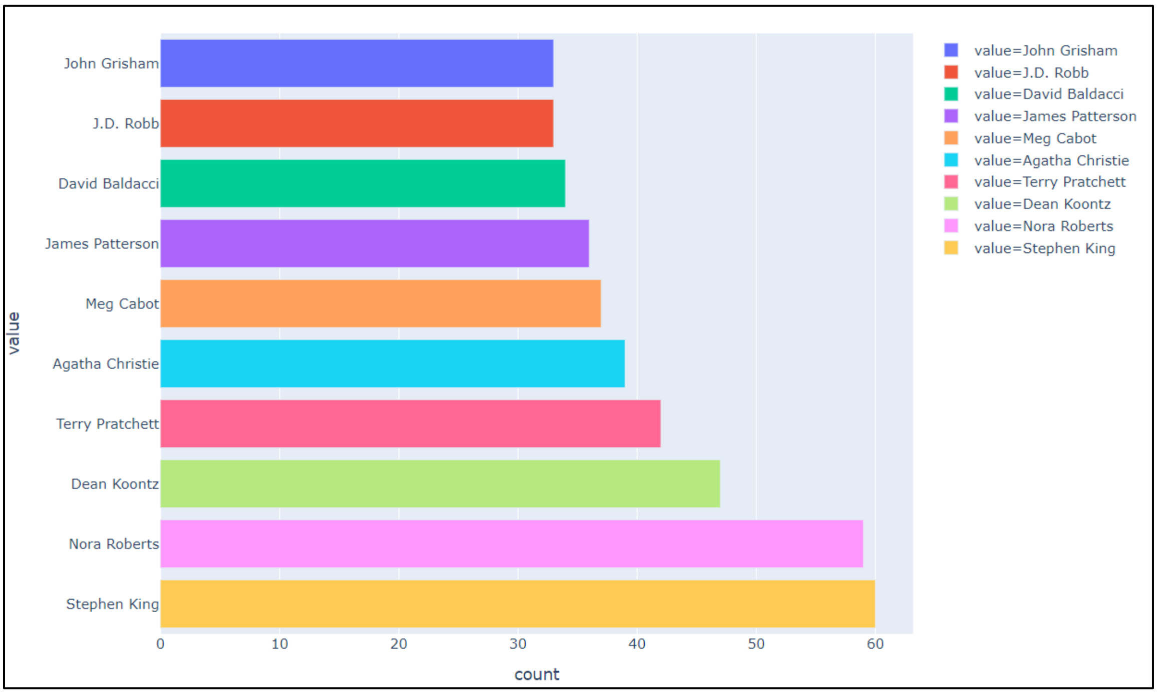


Figure 18:EDA-Graph-2

Which year has maximum number of books published

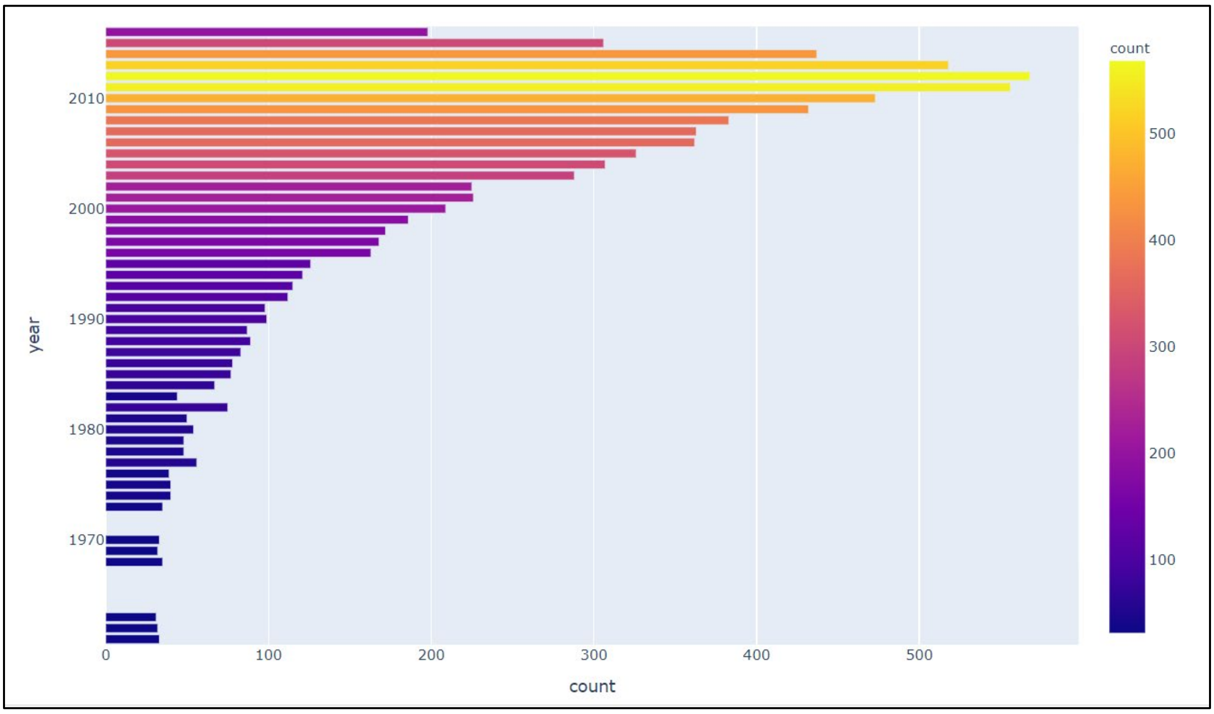


Figure 19:EDA-Graph-3

Which language books are more

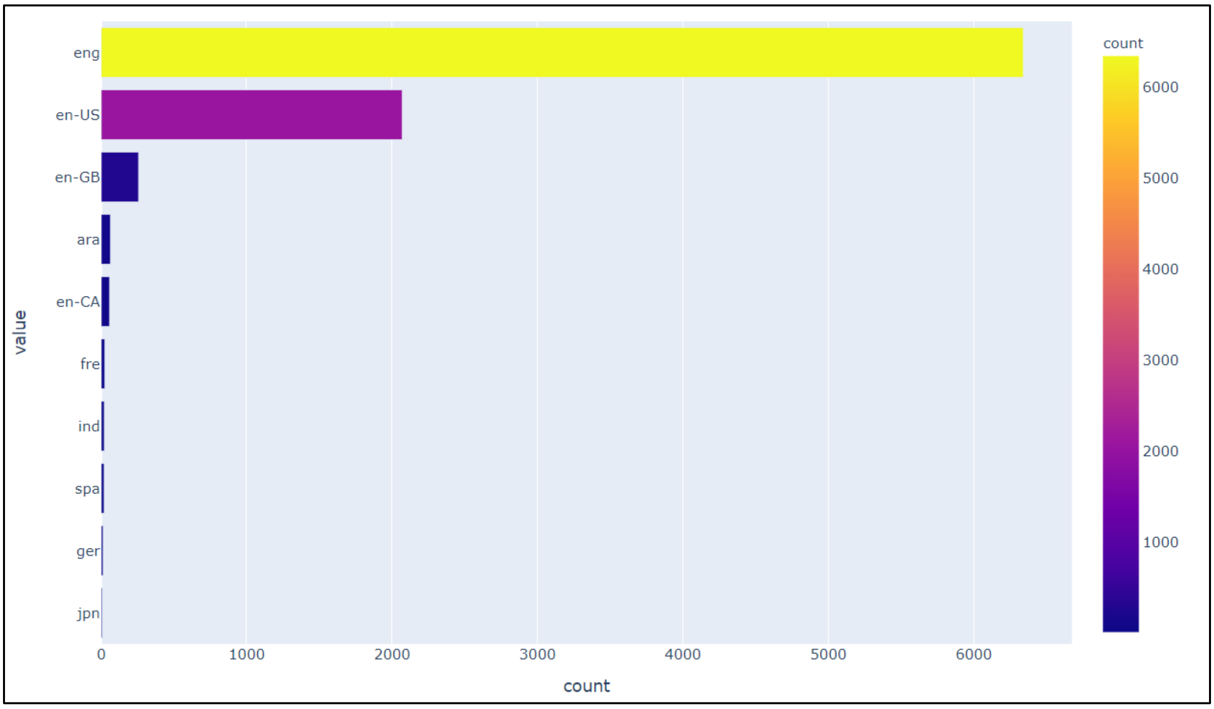


Figure 20:EDA-Graph-4

WordCloud for Book Title

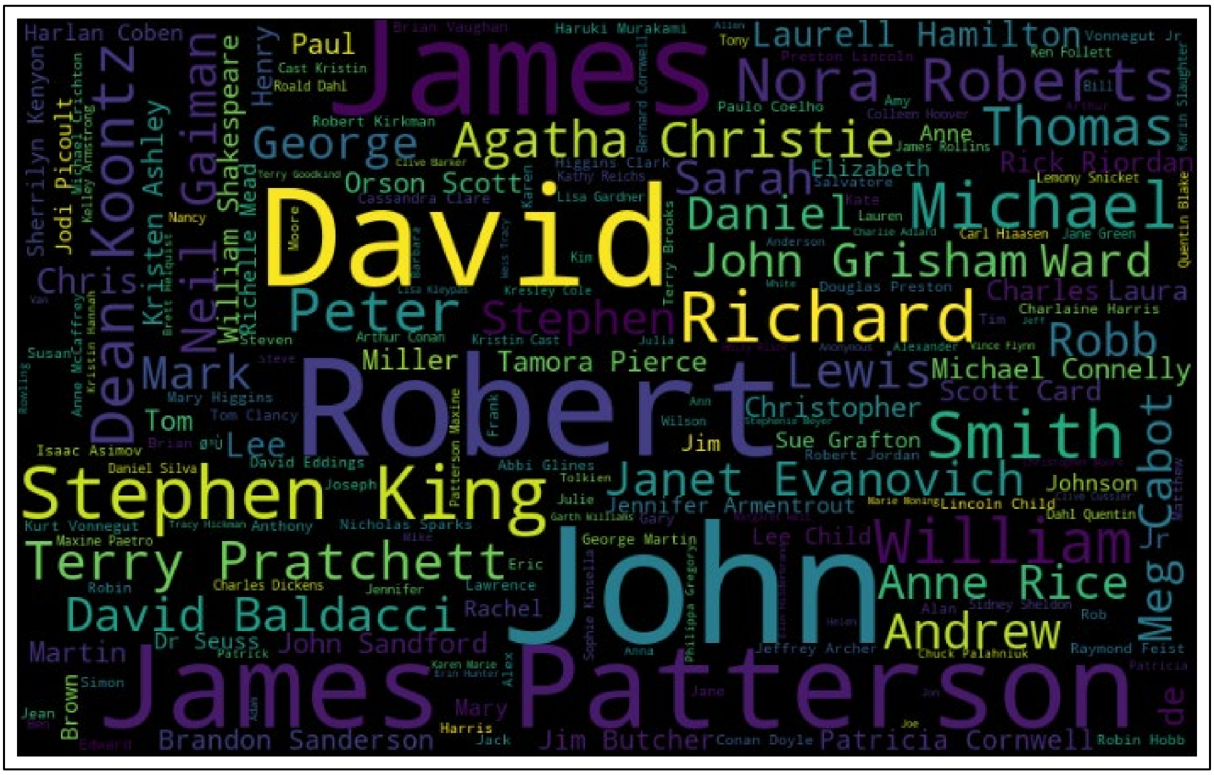


Figure 21:EDA-Graph-5

WordCloud for Book Title

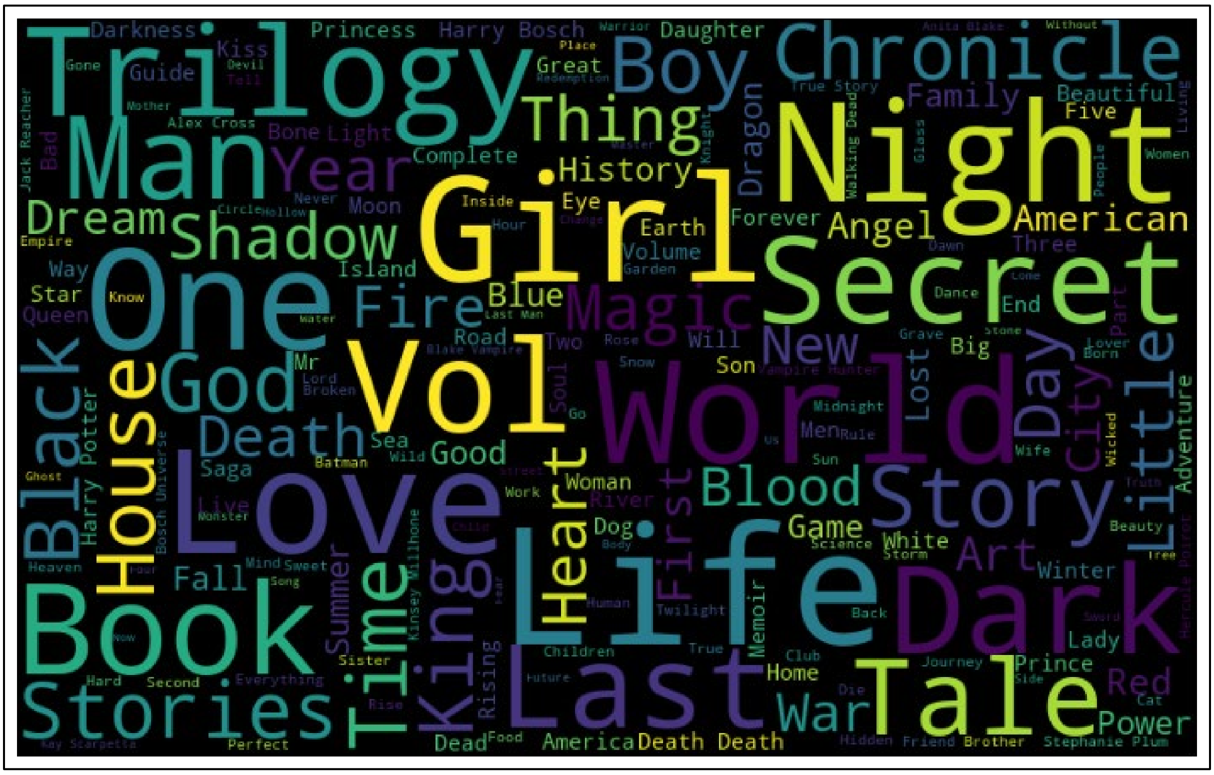


Figure 22:EDA-Graph-6

WebScrapping.py:

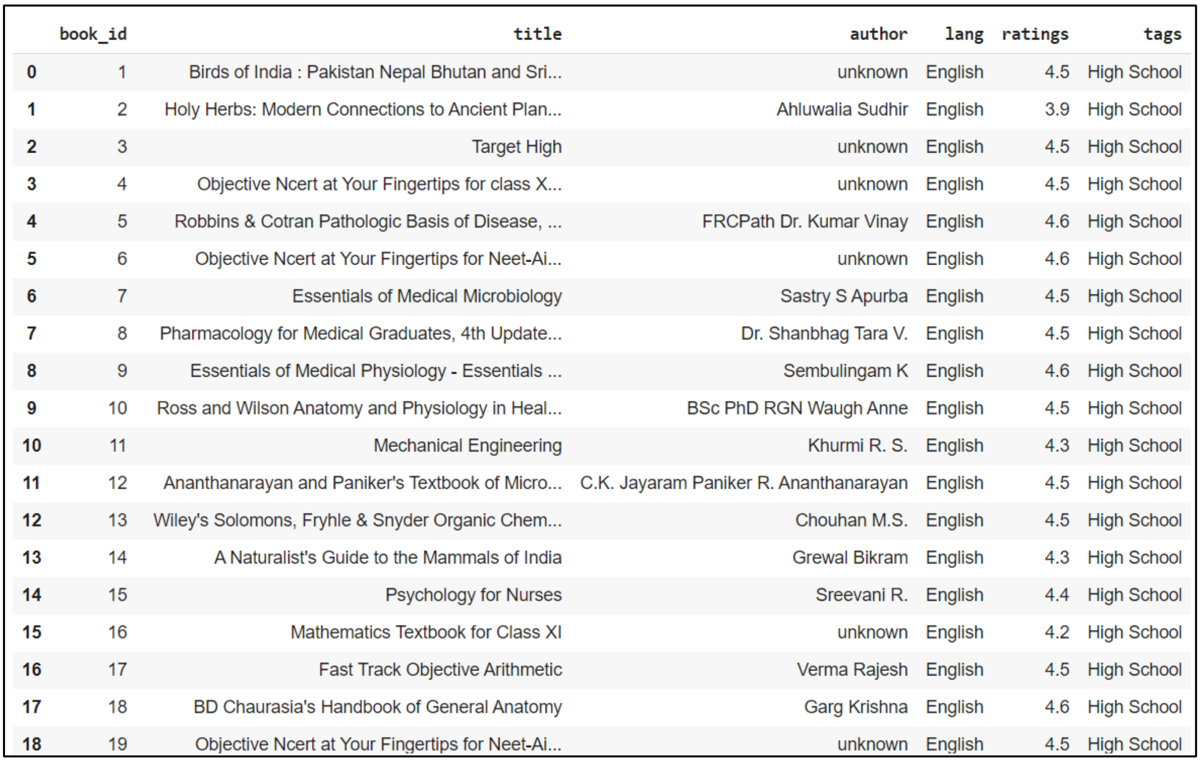
```
### ****************** WEBSCRAPPING.PY ********************
# **WEB SCRAPPING**
### **PRE-REQUSITES**
!pip install beautifulsoup
!pip install selenium
!pip install requests

from selenium import webdriver
from bs4 import BeautifulSoup
import pandas as pd
import requests

"""#**SAMPLE**"""
url="https://www.flipkart.com/search?q=laptop&sid=6bo%2Cb5g&as=on&as-show=on&otracker=AS_QueryStore_OrganicAutoSuggest_1_6_na_na_na&otracker1=AS_QueryStore_OrganicAutoSuggest_1_6_na_na_na&as-pos=1&as-type=RECENT&suggestionId=laptop%7CLaptops&requestId=7ec220e8-4f02-4150-9e0b-9e90cf692f4b&as-searchtext=laptop"
response = requests.get(url)
htmlcontent = response.content
soup = BeautifulSoup(htmlcontent,"html.parser")

print(soup.prettify)

products=[]
prices=[]
ratings=[]
product=soup.find('div',attrs={'class':'_4rR01T'})
print(product.text)

for a in soup.findAll('a',href=True, attrs={'class':'_1fQZEK'}):
  name=a.find('div',attrs={'class':'_4rR01T'})
  price=a.find('div',attrs={'class':'_30jeq3 _1_WHN1'})
  #rating=a.find('div',attrs={'class':'_3LWZlK'})
  products.append(name.text)
  prices.append(price.text)
  #ratings.append(rating.text)
```

```
products

prices

"""#**MY SECTION**
### **Setting-up URL and variables**
url = "https://www.flipkart.com/books/higher-education-and-professional-books/pr?sid=bks%2Cf50&otracker=categorytree&"
response = requests.get(url)
htmlcontent = response.content
soup = BeautifulSoup(htmlcontent,"html.parser")

#print(soup.prettify)

book_id = []
title = []
author = []
ratings = []
lang = []

title1 = soup.find('a',attrs={'class':'s1Q9rs'})
print(title1.text)

"""### **Book ID**"""
book_id = []

m = 1
while (m <= 40):
  book_id.append(m)
  m=m+1

print(book_id)
len(book_id)

"""###  **Book Title**"""
title = []

#extracting book title
```

```
title1 = soup.find_all('a', class_ = 's1Q9rs')
for i in range(len(title1)):
  title.append(title1[i].text)

print(title)
len(title)

"""### **Book Rating**"""
#extracting ratings
rating1 = soup.find_all('div', class_ = '_3LWZlK')
for i in range(len(title1)):
  ratings.append(rating1[i].text)

print(ratings)
len(ratings)

"""### **Book Author + Language + Format**"""
#extracting all the author + language + format
author1 = soup.find_all('div', class_ = '_3Djpdu')
for i in range(len(author1)):
  author.append(author1[i].text)

print(author)
len(author)

#run only you want to reset the list
lang = []
#newlist = []

"""**List to seprate values : Author + Langauge + Format**"""

newlist = []
for word in author:
    word = word.split(",")
    newlist.extend(word)

#checking all raw columns are in right format or not
from tabulate import tabulate
print(tabulate(author))
print(len(newlist))

#deleting extra values which ar distracting the format of tabulated list
bcz of sepration by comma
#run twice to make proper format
#-----------------
#del newlist[14]
#-----------------

print(newlist[14])
print(newlist)
print(len(newlist))

print(author)
print(newlist)
print(len(newlist))

"""**Splitting lang here**"""
i = 1
j = 0
while (i <= len(newlist)/3):
  #print(newlist[j])
  lang.append(newlist[j])
  j = j + 3
  i = i+1

print(lang)
print(len(lang))

"""**Splitting Author here**"""
new_author = []

i = 1
j = 2
while (i <= len(newlist)/3):
  #print(newlist[j])
  new_author.append(newlist[j])
  j = j + 3
  i = i+1
```

```
print(new_author)
print(len(new_author))
"""###**Creating Tags**"""
#creating tags from high school section
tags = []
tag = 'High School'
i = 1
while (i <= 40):
  tags.append(tag)
  i = i+1
print(tags)
print(len(tags))
"""### **Merging all Values & Creating Dataset**"""
df = {'book_id':book_id, 'title':title, 'author': new_author, 'lang': lang, 'ratings': ratings, 'tags': tags}
dataset = pd.DataFrame(data=df)
dataset
```

Output:



	book_id	title	author	lang	ratings	tags
0	1	Birds of India : Pakistan Nepal Bhutan and Sri...	unknown	English	4.5	High School
1	2	Holy Herbs: Modern Connections to Ancient Plan...	Ahluwalia Sudhir	English	3.9	High School
2	3	Target High	unknown	English	4.5	High School
3	4	Objective Ncert at Your Fingertips for class X...	unknown	English	4.5	High School
4	5	Robbins & Cotran Pathologic Basis of Disease, ...	FRCPath Dr. Kumar Vinay	English	4.6	High School
5	6	Objective Ncert at Your Fingertips for Neet-Ai...	unknown	English	4.6	High School
6	7	Essentials of Medical Microbiology	Sastry S Apurba	English	4.5	High School
7	8	Pharmacology for Medical Graduates, 4th Update...	Dr. Shanbhag Tara V.	English	4.5	High School
8	9	Essentials of Medical Physiology - Essentials ...	Sembulingam K	English	4.6	High School
9	10	Ross and Wilson Anatomy and Physiology in Heal...	BSc PhD RGN Waugh Anne	English	4.5	High School
10	11	Mechanical Engineering	Khurmi R. S.	English	4.3	High School
11	12	Ananthanarayan and Paniker's Textbook of Micro...	C.K. Jayaram Paniker R. Ananthanarayan	English	4.5	High School
12	13	Wiley's Solomons, Fryhle & Snyder Organic Chem...	Chouhan M.S.	English	4.5	High School
13	14	A Naturalist's Guide to the Mammals of India	Grewal Bikram	English	4.3	High School
14	15	Psychology for Nurses	Sreevani R.	English	4.4	High School
15	16	Mathematics Textbook for Class XI	unknown	English	4.2	High School
16	17	Fast Track Objective Arithmetic	Verma Rajesh	English	4.5	High School
17	18	BD Chaurasia's Handbook of General Anatomy	Garg Krishna	English	4.6	High School
18	19	Objective Ncert at Your Fingertips for Neet-Ai...	unknown	English	4.5	High School

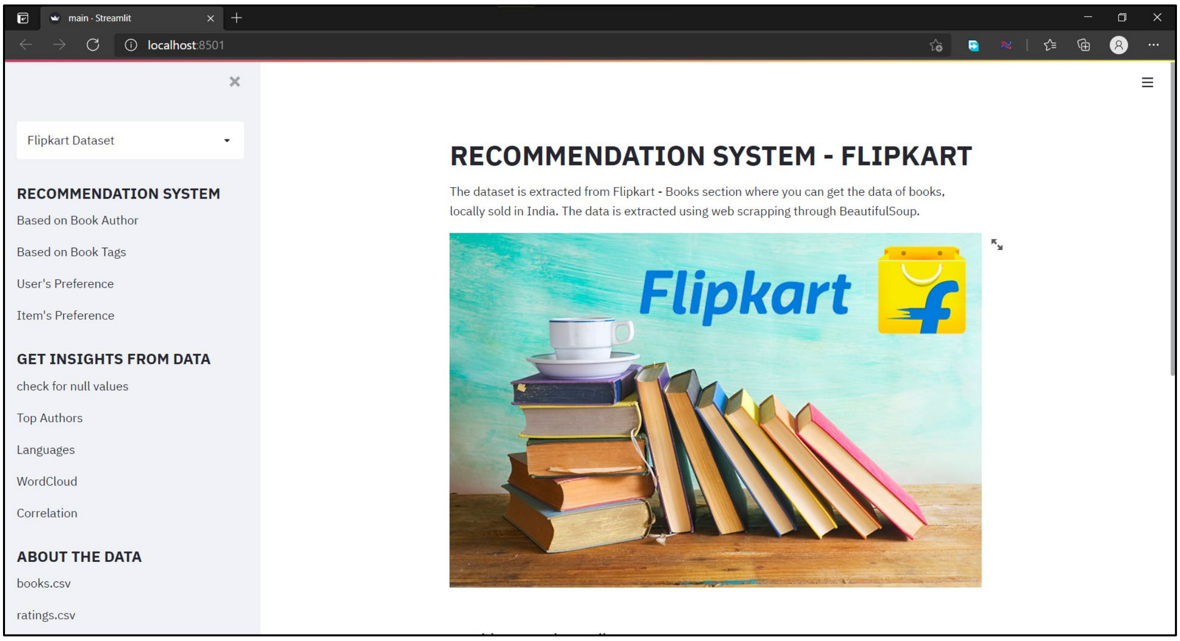
Figure 23:Flipkart-dataset

CHAPTER 5

RESULTS AND DISCUSSION

In this chapter we shall see what results have we achieved form the experiments and implementations of above. Here are some of the snippets of above experiment and final results which demonstrates how recommendation system is fruitful.

Flipkart Page:

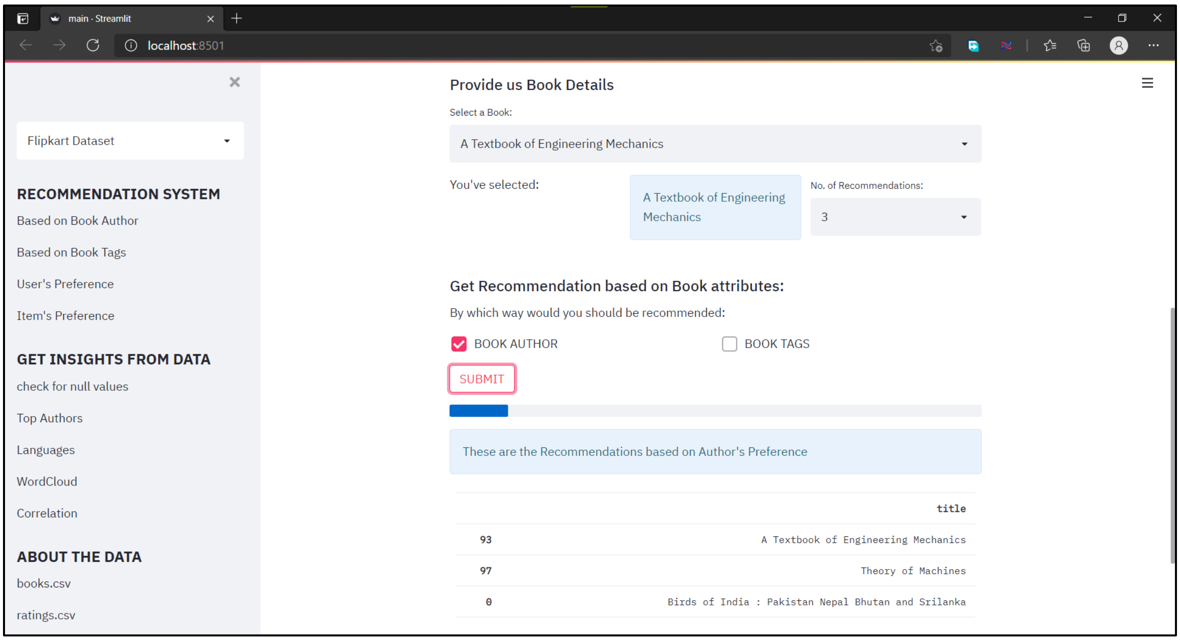


RECOMMENDATION SYSTEM - FLIPKART

The dataset is extracted from Flipkart - Books section where you can get the data of books, locally sold in India. The data is extracted using web scrapping through BeautifulSoup.

Figure 24:Flipkart-page1

Author based:



Provide us Book Details

Select a Book: A Textbook of Engineering Mechanics

You've selected: A Textbook of Engineering Mechanics

No. of Recommendations: 3

Get Recommendation based on Book attributes:

By which way would you should be recommended:

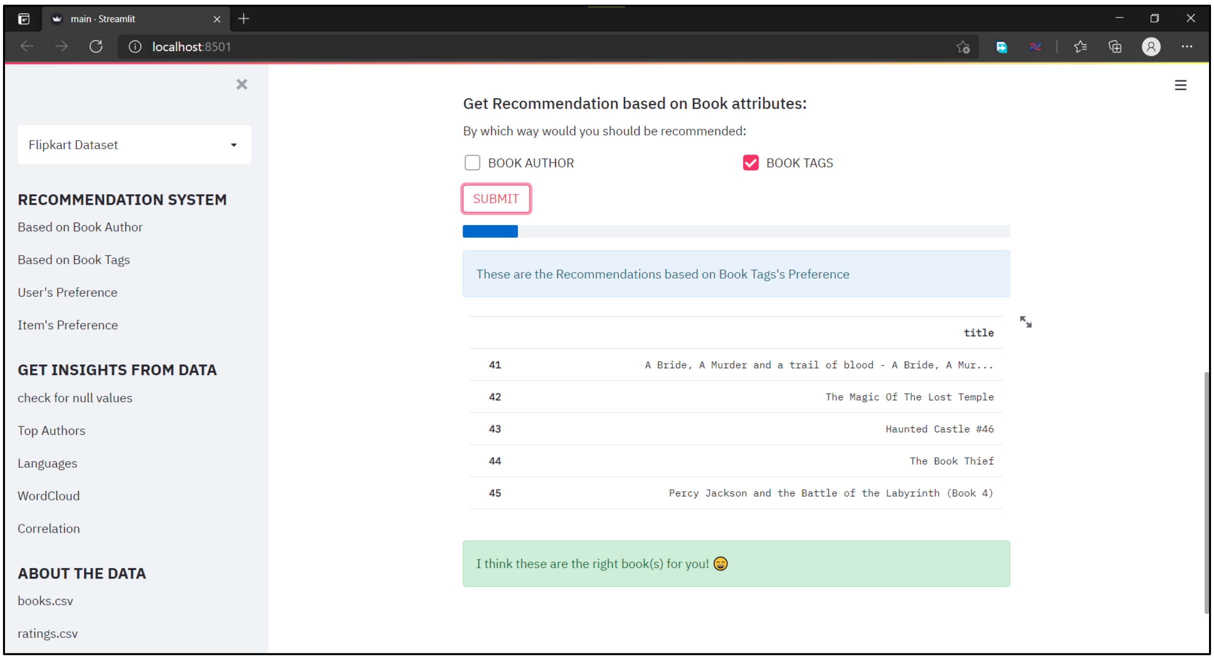
☑ BOOK AUTHOR ☐ BOOK TAGS

These are the Recommendations based on Author's Preference

	title
93	A Textbook of Engineering Mechanics
97	Theory of Machines
0	Birds of India : Pakistan Nepal Bhutan and Srilanka

Figure 25:Flipkart-Author

Tags based:



Get Recommendation based on Book attributes:

By which way would you should be recommended:

☐ BOOK AUTHOR ☑ BOOK TAGS

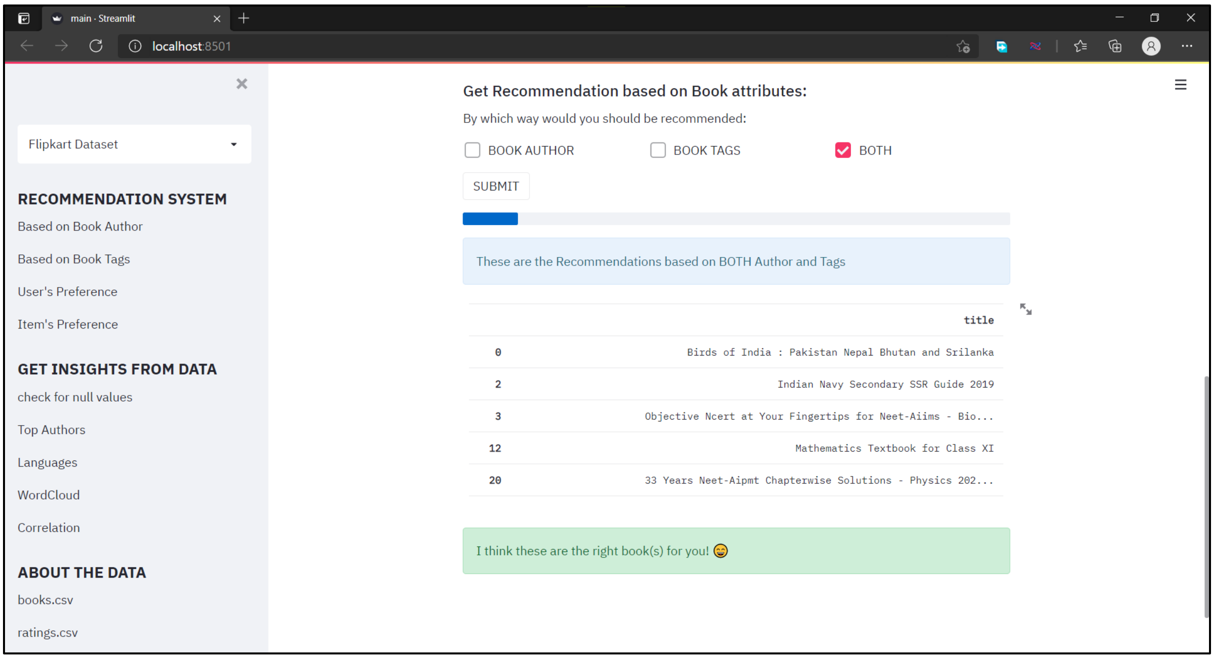
These are the Recommendations based on Book Tags's Preference

	title
41	A Bride, A Murder and a trail of blood - A Bride, A Mur...
42	The Magic Of The Lost Temple
43	Haunted Castle #46
44	The Book Thief
45	Percy Jackson and the Battle of the Labyrinth (Book 4)

I think these are the right book(s) for you! 😄

Figure 26:Flipkart-Tags

Both Author and Tag based:



Get Recommendation based on Book attributes:

By which way would you should be recommended:

☐ BOOK AUTHOR ☐ BOOK TAGS ☑ BOTH

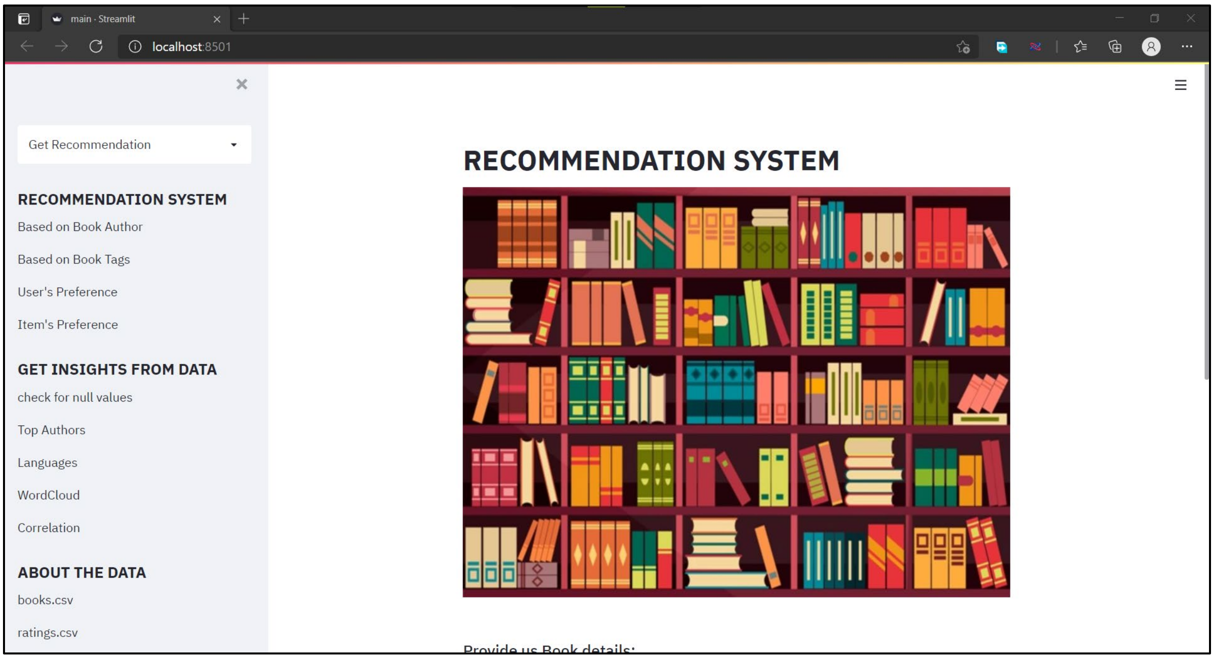
These are the Recommendations based on BOTH Author and Tags

	title
0	Birds of India : Pakistan Nepal Bhutan and Srilanka
2	Indian Navy Secondary SSR Guide 2019
3	Objective Ncert at Your Fingertips for Neet-Aiims - Bio...
12	Mathematics Textbook for Class XI
20	33 Years Neet-Aipmt Chapterwise Solutions - Physics 202...

I think these are the right book(s) for you! 😄

Figure 27:Flipkart-Both

GoodReads Page:

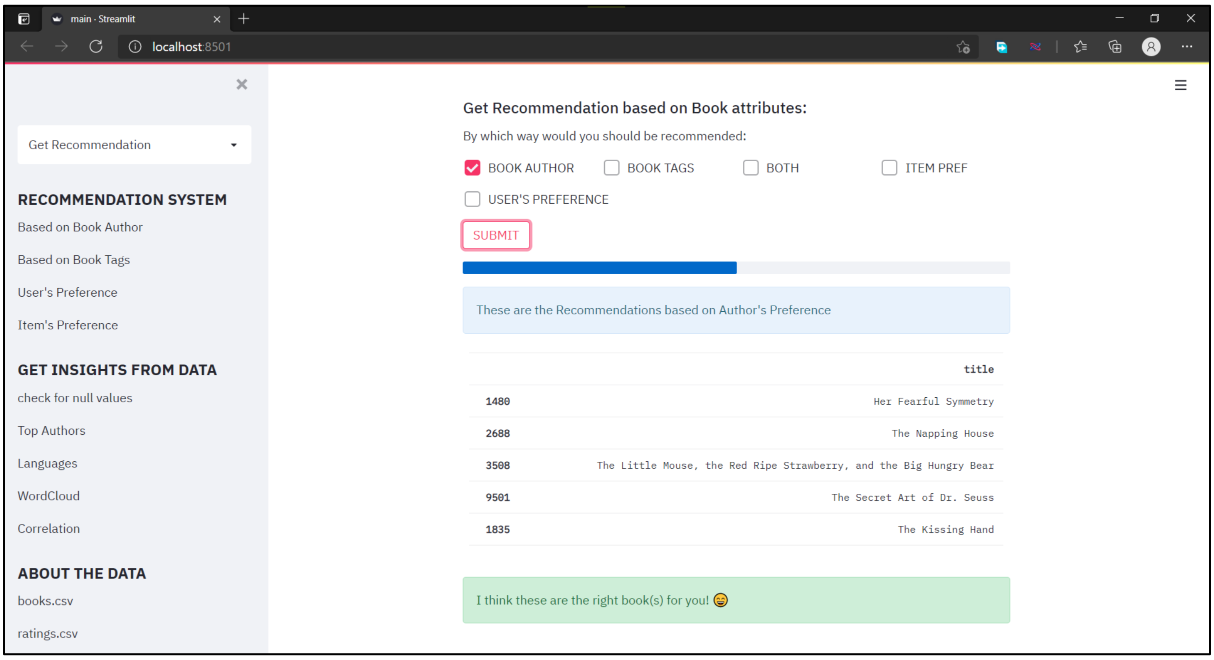


RECOMMENDATION SYSTEM

Provide us Book details:

Figure 28:GoodReads-page

Author based:



Get Recommendation based on Book attributes:

By which way would you should be recommended:

☑ BOOK AUTHOR ☐ BOOK TAGS ☐ BOTH ☐ ITEM PREF

☐ USER'S PREFERENCE

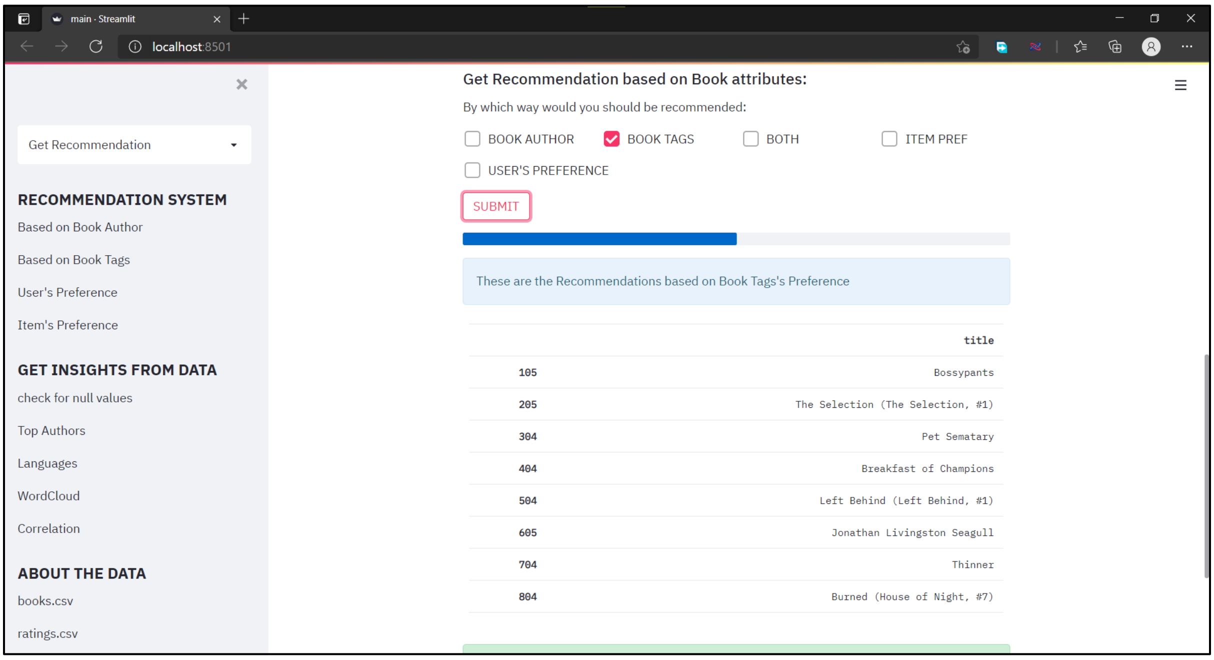
These are the Recommendations based on Author's Preference

	title
1480	Her Fearful Symmetry
2688	The Napping House
3508	The Little Mouse, the Red Ripe Strawberry, and the Big Hungry Bear
9501	The Secret Art of Dr. Seuss
1835	The Kissing Hand

I think these are the right book(s) for you! 😄

Figure 29:GoodReads-author

Tags based:



Get Recommendation based on Book attributes:

By which way would you should be recommended:

□ BOOK AUTHOR ☑ BOOK TAGS □ BOTH □ ITEM PREF

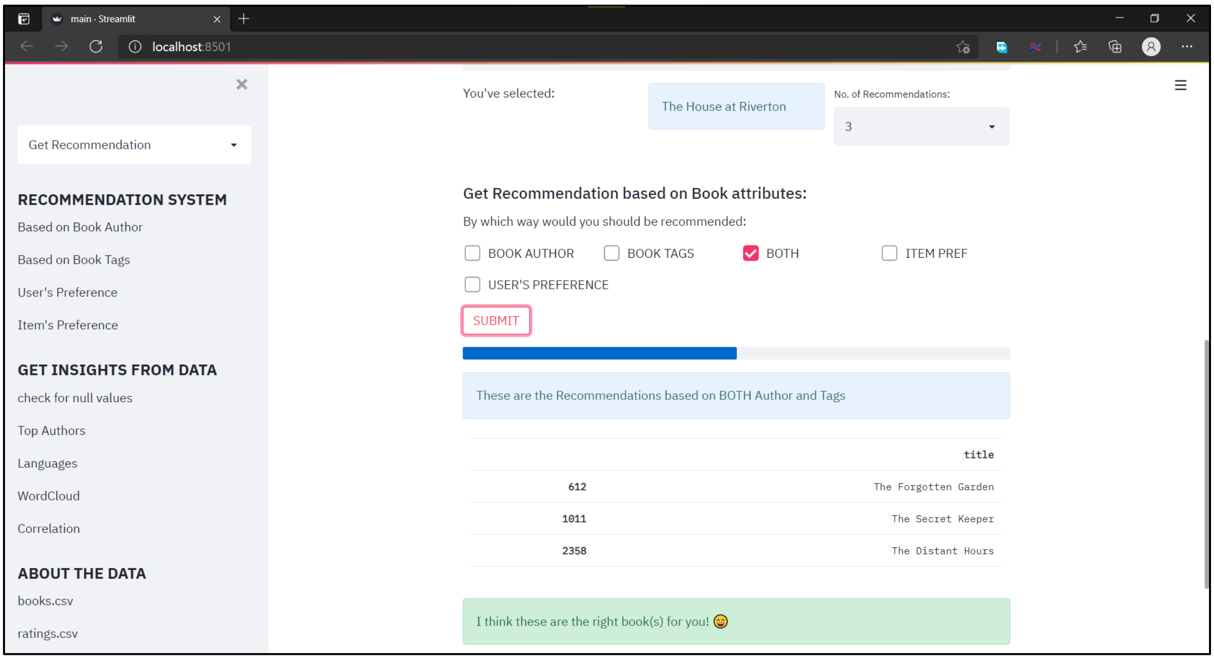
□ USER'S PREFERENCE

These are the Recommendations based on Book Tags's Preference

	title
105	Bossypants
205	The Selection (The Selection, #1)
304	Pet Sematary
404	Breakfast of Champions
504	Left Behind (Left Behind, #1)
605	Jonathan Livingston Seagull
704	Thinner
804	Burned (House of Night, #7)

Figure 30:GoodReads-tags

Both Author and Tag based:



You've selected: The House at Riverton

No. of Recommendations: 3

Get Recommendation based on Book attributes:

By which way would you should be recommended:

□ BOOK AUTHOR □ BOOK TAGS ☑ BOTH □ ITEM PREF

□ USER'S PREFERENCE

These are the Recommendations based on BOTH Author and Tags

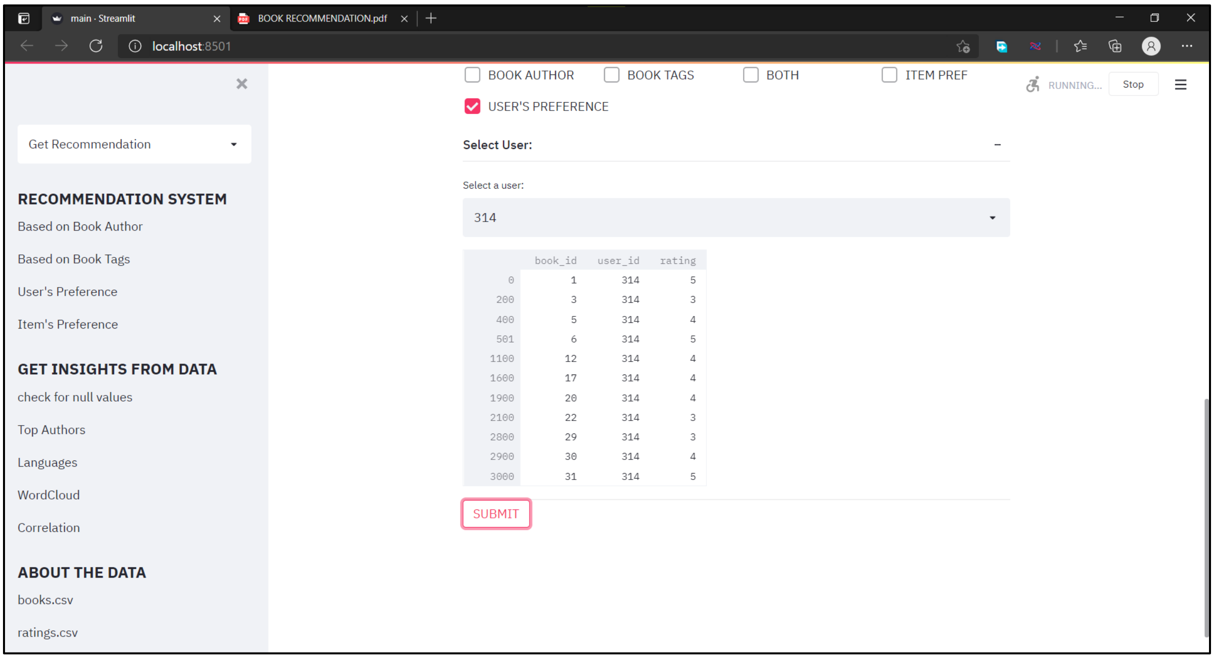
	title
612	The Forgotten Garden
1011	The Secret Keeper
2358	The Distant Hours

I think these are the right book(s) for you! 😀

Figure 31:GoodReads-both

User’s Preference:

Selecting user



□ BOOK AUTHOR □ BOOK TAGS □ BOTH □ ITEM PREF

☑ USER'S PREFERENCE

Select User:

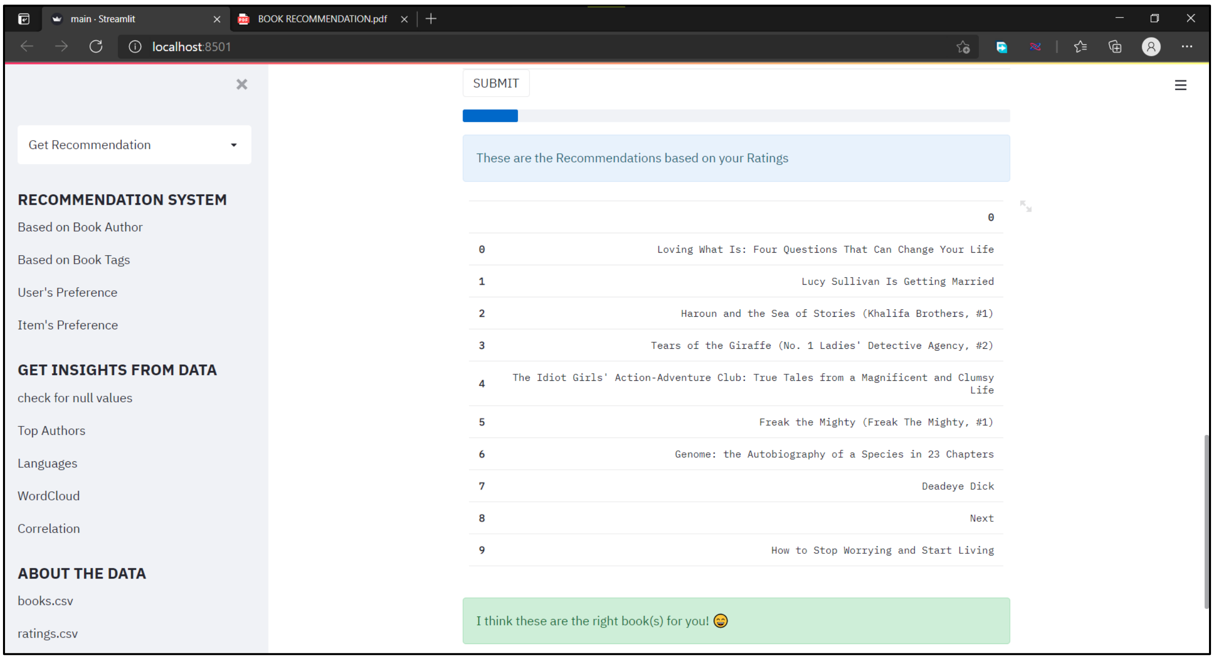
Select a user: 314

	book_id	user_id	rating
0	1	314	5
200	3	314	3
400	5	314	4
501	6	314	5
1100	12	314	4
1600	17	314	4
1900	20	314	4
2100	22	314	3
2800	29	314	3
2900	30	314	4
3000	31	314	5

SUBMIT

Figure 32:GoodReads-user(a)

Getting results



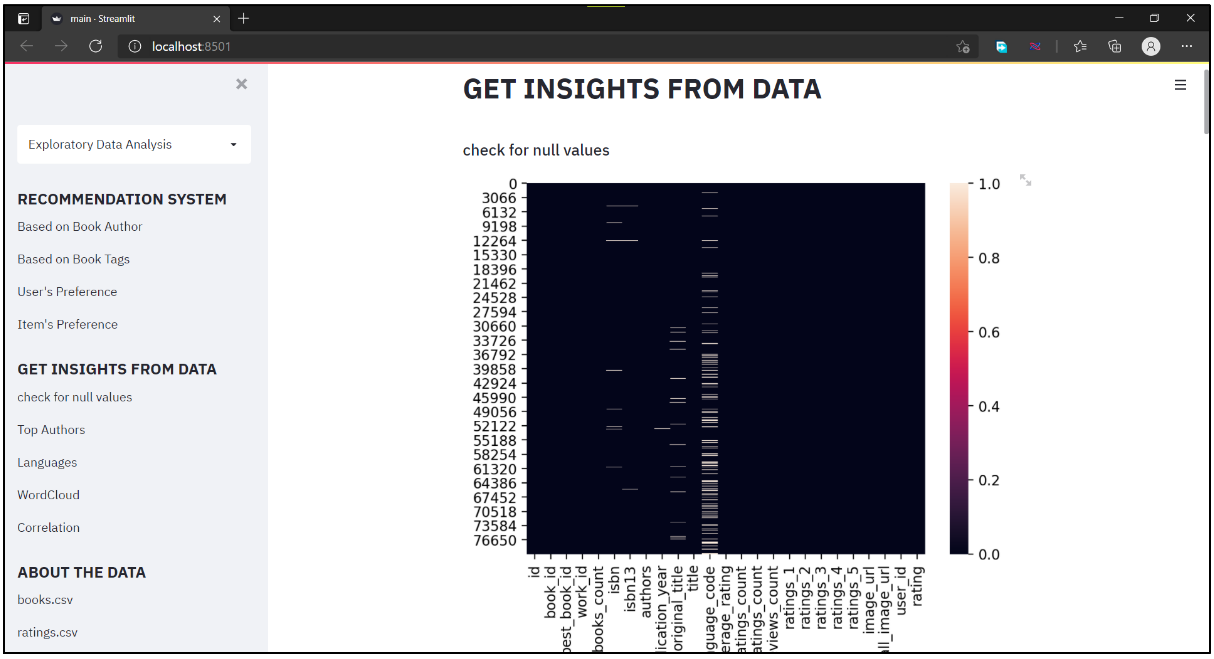
These are the Recommendations based on your Ratings

	0
0	Loving What Is: Four Questions That Can Change Your Life
1	Lucy Sullivan Is Getting Married
2	Haroun and the Sea of Stories (Khalifa Brothers, #1)
3	Tears of the Giraffe (No. 1 Ladies' Detective Agency, #2)
4	The Idiot Girls' Action-Adventure Club: True Tales from a Magnificent and Clumsy Life
5	Freak the Mighty (Freak The Mighty, #1)
6	Genome: the Autobiography of a Species in 23 Chapters
7	Deadeye Dick
8	Next
9	How to Stop Worrying and Start Living

I think these are the right book(s) for you! 😀

Figure 33:GoodReads-user(b)

EDA Page:



GET INSIGHTS FROM DATA

check for null values

Figure 34:GoodReads-EDA-1

Count of Ratings:

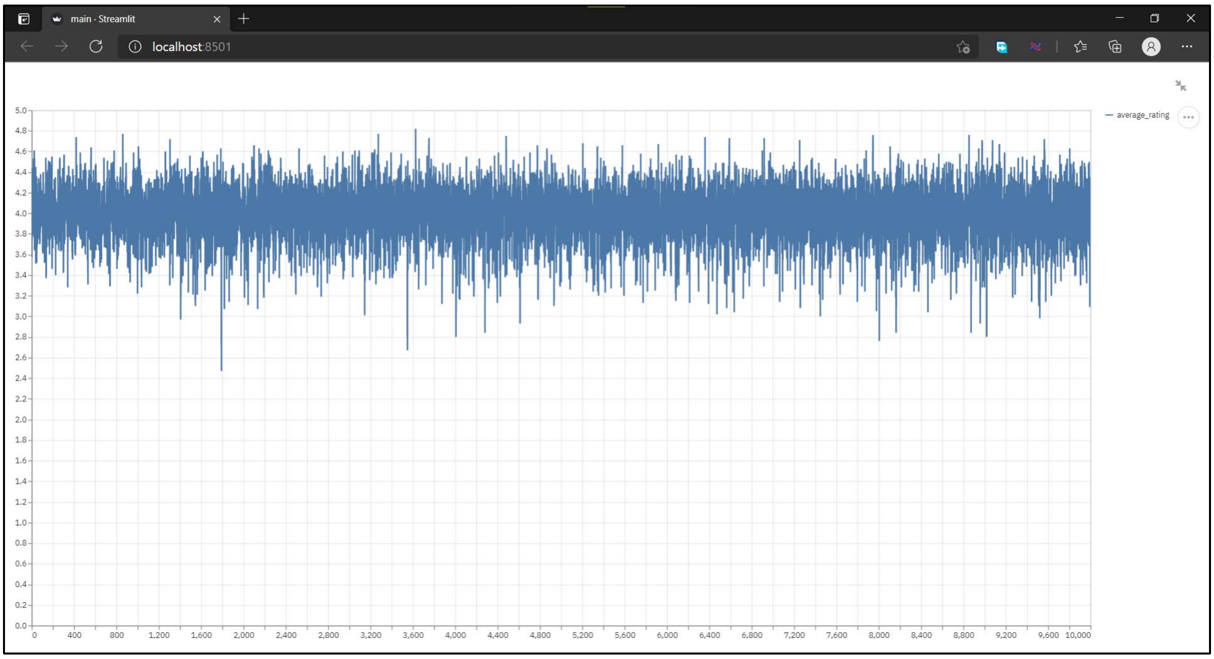
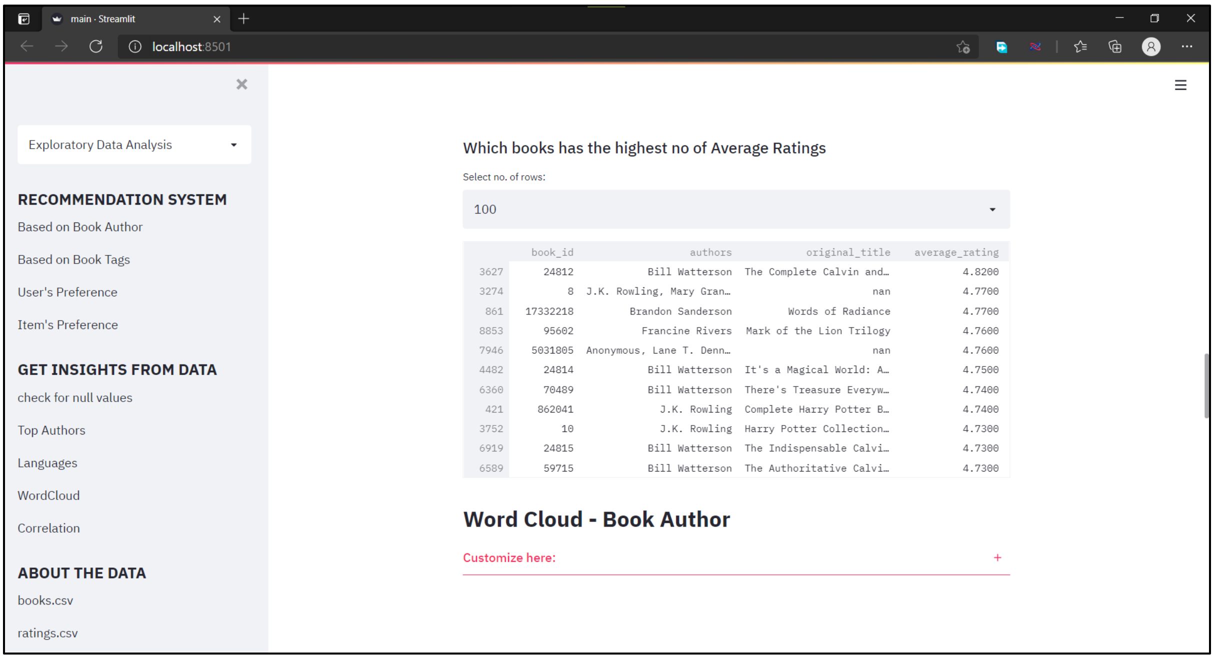


Figure 35:GoodReads-EDA-2

Book having the highest no of Average Ratings:



Which books has the highest no of Average Ratings

Select no. of rows:

100

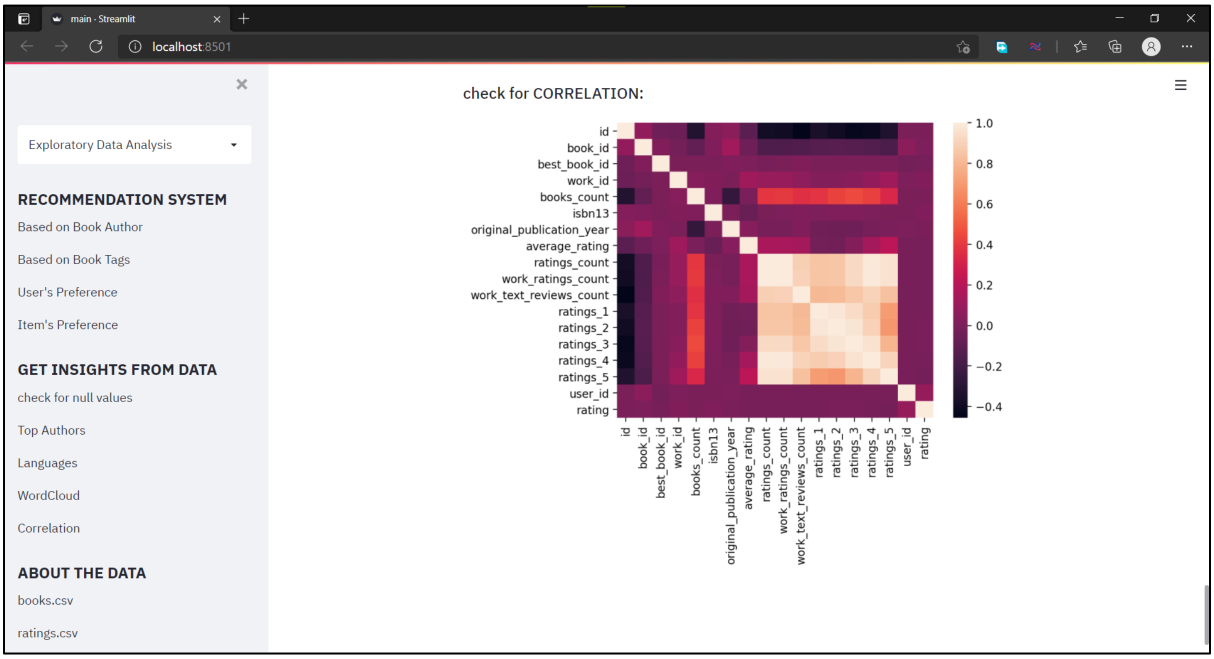
	book_id	authors	original_title	average_rating
3627	24812	Bill Watterson	The Complete Calvin and…	4.8200
3274	8	J.K. Rowling, Mary Gran…	nan	4.7700
861	17332218	Brandon Sanderson	Words of Radiance	4.7700
8853	95602	Francine Rivers	Mark of the Lion Trilogy	4.7600
7946	5031805	Anonymous, Lane T. Denn…	nan	4.7600
4482	24814	Bill Watterson	It's a Magical World: A…	4.7500
6360	70489	Bill Watterson	There's Treasure Everyw…	4.7400
421	862041	J.K. Rowling	Complete Harry Potter B…	4.7400
3752	10	J.K. Rowling	Harry Potter Collection…	4.7300
6919	24815	Bill Watterson	The Indispensable Calvi…	4.7300
6589	59715	Bill Watterson	The Authoritative Calvi…	4.7300

Word Cloud - Book Author

Customize here: +

Figure 36:GoodReads-EDA-3

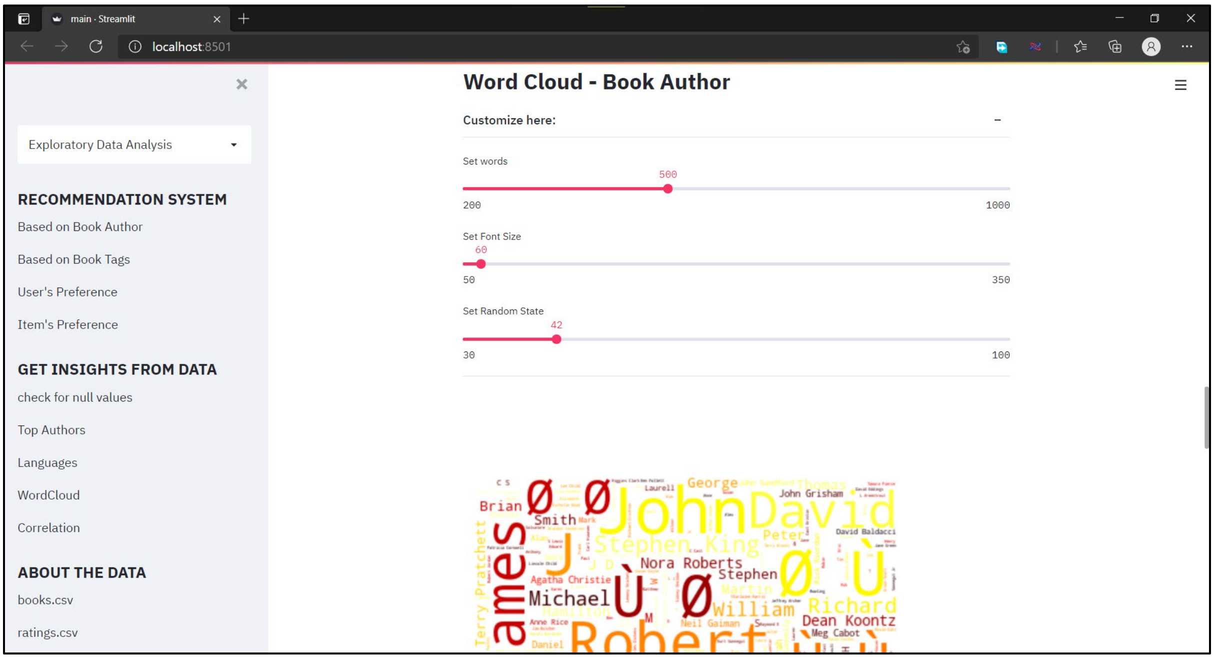
Correlation:



check for CORRELATION:

Figure 37:GoodReads-EDA-6

WordCloud:



Word Cloud - Book Author

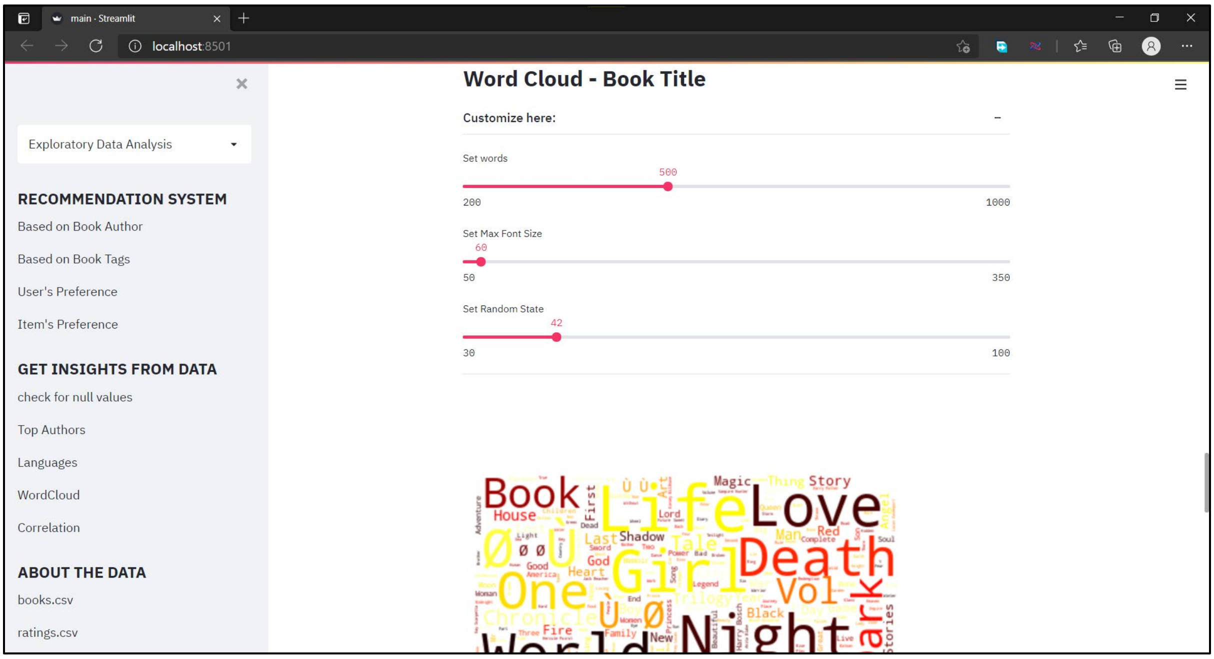
Customize here:

Set words: 500 (200–1000)

Set Font Size: 60 (50–350)

Set Random State: 42 (30–100)

Figure 38:GoodReads-EDA-4



Word Cloud - Book Title

Customize here:

Set words: 500 (200–1000)

Set Max Font Size: 60 (50–350)

Set Random State: 42 (30–100)

Figure 39:GoodReads-EDA-5

CHAPTER 6

CONCLUSION AND PROPOSALS FOR FUTURE WORK

CONCLUSION

Hence, we could conclude that based on correlation cosine similarity and Karl Pearson similarity using TFID vectorization almost both the approaches i.e. content based and collaborative base filtering by different ways on different columns of dataset has finally been achieved. The KNN Basic found suitable for the recommendation and gave favorable results among the others KNN family of Algorithms. While getting the results, please students were merely satisfied with the response of recommended book, which were useful to them. With this we could say that:

- The recommendation has been achieved on the basis of Cosine, Karl Pearson similarity and KNN basic algorithm.
- The results of the recommended books were tested with the students and the recommendation were found fruitful to them.
- The technique of web scrapping covers a wide range of books which helps in recommendation of books more precise and suitable for every user.
- The clean, simple, system UI has made more users to get interacted easily and has all types of users.
- Rather than using a traditional database a run time database extracted from web scrapping has helped the system ticket more accurate output and cover almost all variety of books.

Thus, for covering all types of users and all types of books the next step of this project is true deployed to the level where a user enters a book name though it might be any book based on any attributes the system should be able to fine the top recommended books for the user.

FUTURE WORK

Future work is the implementing of the project to the next level of possibilities what the system can do. The final aim of the project has not completely been achieved. A recommendation should be platform independent database independent and should provide the recommended book even for the new users. This will help the user who is starting with the new topic and searching for a book can get the help and Recommendation so be proposed system and make the write the decision to read the book. As this Technology needs to be platform independent and. Database the dependent the only way is through use web scraping to get the value samples of data on which we can apply the recommendation algorithms.

The future work seems to be interesting and amazing but that's the most crucial and difficult task as being a student

CHAPTER 7

BIBLOGRAPHY

These a few sources through which the central idea of the project arise and also helped me to develop a recommendation system. Following are the articles and few people who are need to include as stakeholder in the current project. I would like my mention my Teacher, Prof. Sagar Kulkani who helped in choosing the right project and supported me while developing dataset from BeautifulSoup Web Scrapping. Also, I would share the creditability to my team, my collogues Mr. Sachin, Ms. Samidha and Mr. Pawan – the Quad Squad team for helping me out in perform Recommendations. Quad Squad for supporting me with ideas. These are the few links and resources which helped me to develop a recommendation system.

- Surprise package for implantation of KNN Basics -
  https://surprise.readthedocs.io/en/stable/similarities.html#module-surprise.similarities

- Dataset has been downloaded from here -
  https://www.goodreads.com/work/editions/2792775-the-hunger-games

- Surprise package to have a look on all other KNN algorithms for recommendations -
  https://surprise.readthedocs.io/en/stable/knn_inspired.html

- StreamLit documentation which helps to develop stunning and responsive user interface for ML domain projects -
  https://docs.streamlit.io/en/stable/api.html#display-progress-and-status

- BeautifulSoup documentation for web scrapping Flipkart Books dataset
  https://www.crummy.com/software/BeautifulSoup/bs4/doc/

- Understanding Cosine Similarity between two vectors using Tfid vectors
  https://towardsdatascience.com/understanding-cosine-similarity-and-its-application-fd42f585296a

- How to convert a name to a Tfid Vectors for feature extraction – explanation part1
  https://scikitlearn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

- How to convert a name to a Tfid Vectors for feature extraction – explanation part 2
  https://medium.com/@cmukesh8688/tf-idf-vectorizer-scikit-learn-dbc0244a911a

- How to implement cosine similarity.
  https://pub.towardsai.net/content-based-recommendation-system-using-word-embeddings-c1c15de1ef95

- A reference on how basic recommendation works

https://analyticsindiamag.com/how-to-build-a-content-based-movie-recommendation-system-in-python/

- Web scrapping
  https://www.edureka.co/blog/web-scraping-with-python/

REFERENCES

- A Hybrid Model for Book Recommendation - Rohit Darekar, Karan Dayma,Rohan Parabh, Prof. Swapnali Kurhade.
- I Nyoman Pande Wahyu Dharmawan, Riyanarto Sarno Informatics - The designof disciplinary book recommendation system based on android
- Book Recommendation using Neo4j Graph Database in BibTeX Book Metadata
- The Design and Implementation of Books Recommendation System
- A Collaborative Filtering Based Library Book Recommendation System
- Deep Learning - Book by Aaron Courville, Ian Goodfellow, and Yoshua Bengio
- Python Machine Learning
- Introduction to Machine Learning with Python: A Guide for Data Scientists - Book by Andreas C. Müller and Sarah Guido

ACKNOWLEDGEMENT

We would like to take this opportunity to take all those individuals who have helped us in visualizing this project.

I express our deep gratitude to our project guide **PROF. ABHIJEET SALVI** forproviding timely assistance with our queries and guidance that he gave owing to his experiencein this field for past many years. He had indeed been a lighthouse for us in this journey.

We extend our appreciation to **GAJANAN WADEKAR** all our Professors from **PILLAI COLLEGE OF ARTS, COMMERCE AND SCIENCE** for their valuable insight and tips during the designing of the project. Their contributions have been valuable in so many ways that we find it difficult to acknowledge of them individual.

I would also thankful to our Head of Dept. **PROF. ABHIJEET SALVI** and **PROF. SAGAR KULKARNI** for extending her help directly and indirectly through various channel in our project work.

SADIQ SHAIKH