“Book Recommendation System”

Capstone Project

Submitted To

AlmaBetter

Submitted By:

Vikas panchal
Email id: - panchalvicky501@gmail.com

Naveen Kumar batta
Email id: - naveenbatta4587@gmail.com

Abstract

As the amounts of online books are exponentially increasing due to COVID-19 pandemic, finding relevant books from a vast e-book space becomes a tremendous challenge for online users. Personal recommendation systems have been emerged to conduct effective search which mine related books based on user rating and interest. Most of these existing systems are user-based ratings where content-based and collaborative-based learning methods are used. These systems' irrationality is their rating technique, which counts the users who have already been unsubscribed from the services and no longer rate books. This paper proposed an effective system for recommending books for online users that rated a book using the clustering method and then found a similarity of that book to suggest a new book. The proposed system used the K-means Cosine Distance function to measure distance and Cosine Similarity function to find Similarity between the book clusters. Sensitivity, Specificity, and F Score were calculated for ten different datasets. The average Specificity was higher than sensitivity, which means that the classifier could re-move boring books from the reader's list. Besides, a receiver operating characteristic curve was plotted to find a graphical view of the classifiers' accuracy. Most of the datasets were close to the ideal diagonal classifier line and far from the worst classifier line. The result concludes that recommendations, based on a particular book, are more accurately effective than a user-based recommendation system.

Table of Contents

Chapter 1. Introduction:

1.0 Introduction:

Most organizations have their recommendation system when they sell products online. But almost all the websites are not developed of the buyer interest; the organizations' force add-on sells to buyers by recommending unnecessary and irrelevant products. A personalized recommendation system (PRS) helps individual users find exciting and useful products from a massive collection of items. With the growth of the internet, consumers have lots of options on products from ecommerce sites. Finding the right products at the right time is a real challenge for consumers. A personalized recommendation system helps users find books, news, movies, music, online courses, and research articles.

Since, here we are trying to recommend books to users based on their past purchases or ratings the user gave previously, we are basically trying different models like Popularity based recommender system, Collaborative filtering-based recommender system (user-item or item-item) etc. We will be using the Popularity based recommender system to deal with the cold start problem, where we do not have history of past purchases of a particular user or where the user is totally new.

1.1 Recommendation System:

Recommendation systems produce a ranked list of items on which a user might be interested, in the context of his current choice of an item.

- ❑ Subclass of Information filtering system that seek to predict the 'rating' or 'preference' that a user would give to them.
- ❑ Applied in variety of applications like movies, books, research articles.
- ❑ Recommendation systems involve predicting user preferences for unseen items • such as movies, songs or books.
- ❑ Recommendation systems have become very popular with the increasing availability of millions of products online.
- ❑ Recommending relevant products increases the sales.

.

1.2 Problem statement:

During the last few decades, with the rise of YouTube, Amazon, Netflix, and many other such web services, recommender systems have taken more and more place in our lives. From e-commerce (suggest to buyers articles that could interest them) to online advertisement (suggest to users the right contents, matching their preferences), recommender systems are today unavoidable in our daily online journeys.

In a very general way, recommender systems are algorithms aimed at suggesting relevant items to users (items being movies to watch, text to read, products to buy, or anything else depending on industries).

The main objective is to create a recommendation system to recommend relevant books to users based on popularity and user interests.

1.3 Objective:

- The main objective is to create a machine learning model to recommend relevant books to users based on popularity and user interests.
- In addition to the ML Model prediction, we also have taken into account the book recommendation for a totally new user.

Chapter 2

2. Data Summary –

2.1 Data Description –

We are using Book-Crossing dataset to train and test our recommendation system. **Book-Crossings** is a book ratings dataset compiled by Cai-Nicolas Ziegler. It contains 1.1 million ratings of 270,000 books by 90,000 users. The ratings are on a scale from 1 to 10. The Book-Crossing dataset comprises 3 files.

● Users : -

This .csv file contains the users. Note that user IDs (User-ID) have been anonymized and map to integers. Demographic data is provided (Location, Age) if available. Otherwise, these fields contain NULL values.

● Books :-

Books are identified by their respective ISBN. Invalid ISBNs have already been removed from the dataset. Moreover, some content-based information is given (Book-Title, Book-Author, Year-Of-Publication, Publisher), obtained from Amazon Web Services. Note that in the case of several authors, only the first is provided. URLs linking to cover images are also given, appearing in three different flavors (Image-URL-S, Image-URL-M, Image-URL-L), i.e., small, medium, large. These URLs point to the Amazon website.

● Ratings :-

Contains the book rating information. Ratings (Book-Rating) are either explicit, expressed on a scale from 1–10 (higher values denoting higher appreciation), or implicit, expressed by 0.

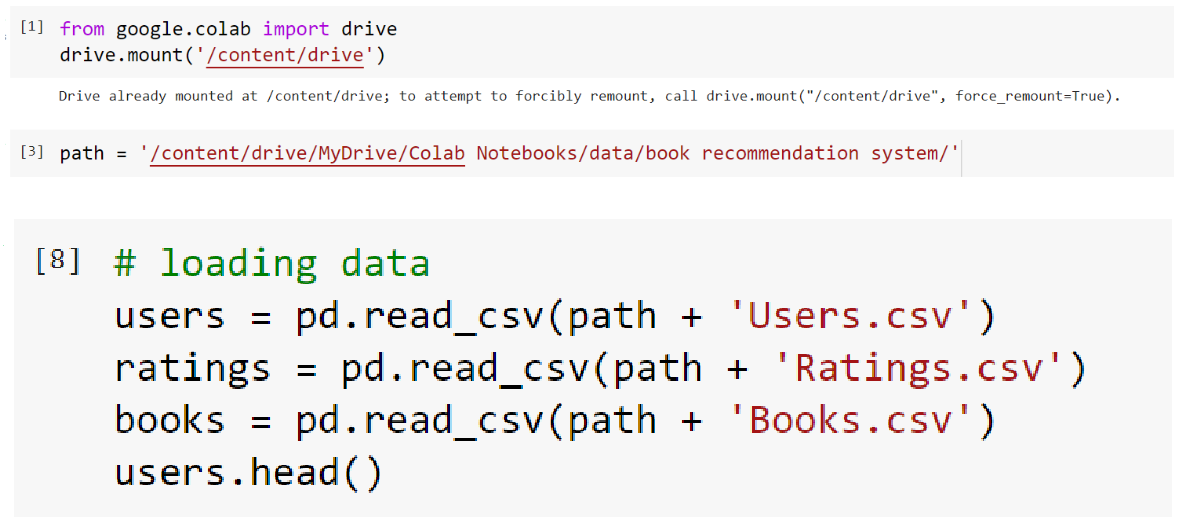
2.2 Import the Dataset –

Before building any machine learning model, it is vital to understand what the data is, and what are we trying to achieve. Data exploration reveals the hidden trends and insights and data pre-processing makes the data ready for use by ML algorithms.

So, let’s begin. . .

To proceed with the problem dealing first we will load our dataset that is given to us in a .csv file into a **data frame**.

Mount the drive and load the csv file into a data frame.



```
[1] from google.colab import drive
    drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[3] path = '/content/drive/MyDrive/Colab Notebooks/data/book recommendation system/'

[8] # loading data
    users = pd.read_csv(path + 'Users.csv')
    ratings = pd.read_csv(path + 'Ratings.csv')
    books = pd.read_csv(path + 'Books.csv')
    users.head()
```

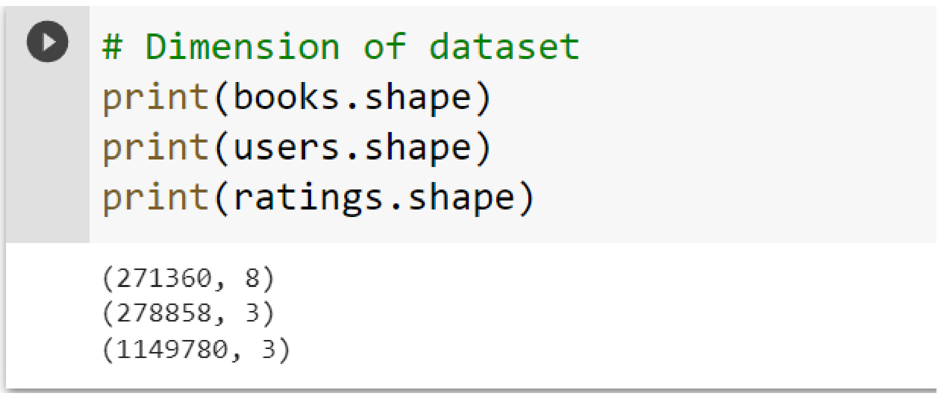
Fig 2. import dataset.

2.3 Exploratory Data Analysis (EDA) –

The primary goal of EDA is to support the analysis of data prior to making any conclusions. It may aid in the detection of apparent errors, as well as a deeper understanding of data patterns, the detection of outliers or anomalous events, and the discovery of interesting relationships between variables.

Duplication: In our further analysis we found that the dataset has no duplicate entries.

2.3.1 Dimension of dataset:



```
# Dimension of dataset
print(books.shape)
print(users.shape)
print(ratings.shape)

(271360, 8)
(278858, 3)
(1149780, 3)
```

2.3.2 Users-Dataset:

In the users-dataset we have the following feature variables.

- User-ID (unique for each user)
- Location (contains city, state and country separated by commas)
- Age

Out of these features, User-ID is unique for each user, Location contains city, state and country separated by commas and we have Age given across each user.

	User-ID	Location	Age
0	1	nyc, new york, usa	NaN
1	2	stockton, california, usa	18.0
2	3	moscow, yukon territory, russia	NaN
3	4	porto, v.n.gaia, portugal	17.0
4	5	farnborough, hants, united kingdom	NaN

Missing Values:- Find the users missing values given dataset.

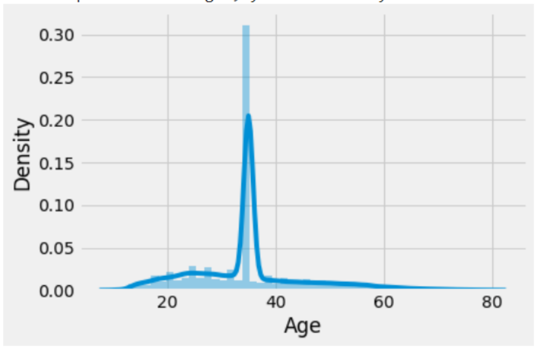
```
[9]  users.isnull().sum()
```

```
User-ID          0
Location         0
Age         110762
dtype: int64
```

Age: Let's understand the Age distribution of the given user dataset. By using a dist-plot for the Age column we can get the distribution of ages and the density. Below is the distplot.

```
sns.distplot(users.Age)
```

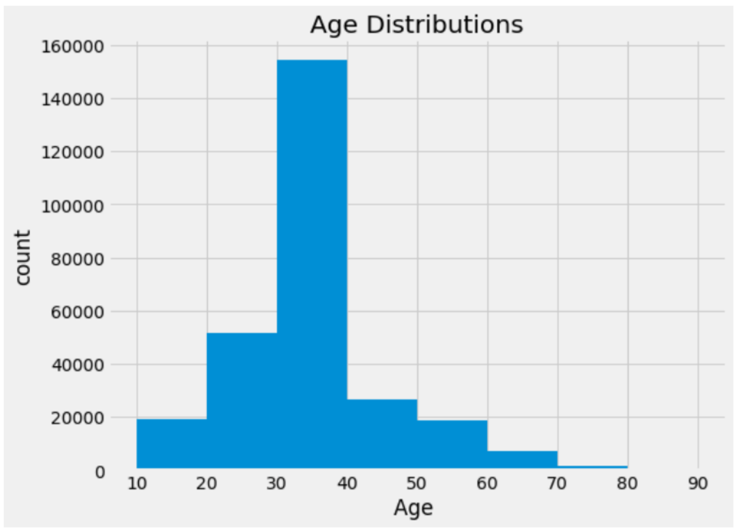
```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619:
  warnings.warn(msg, FutureWarning)
<AxesSubplot:xlabel='Age', ylabel='Density'>
```



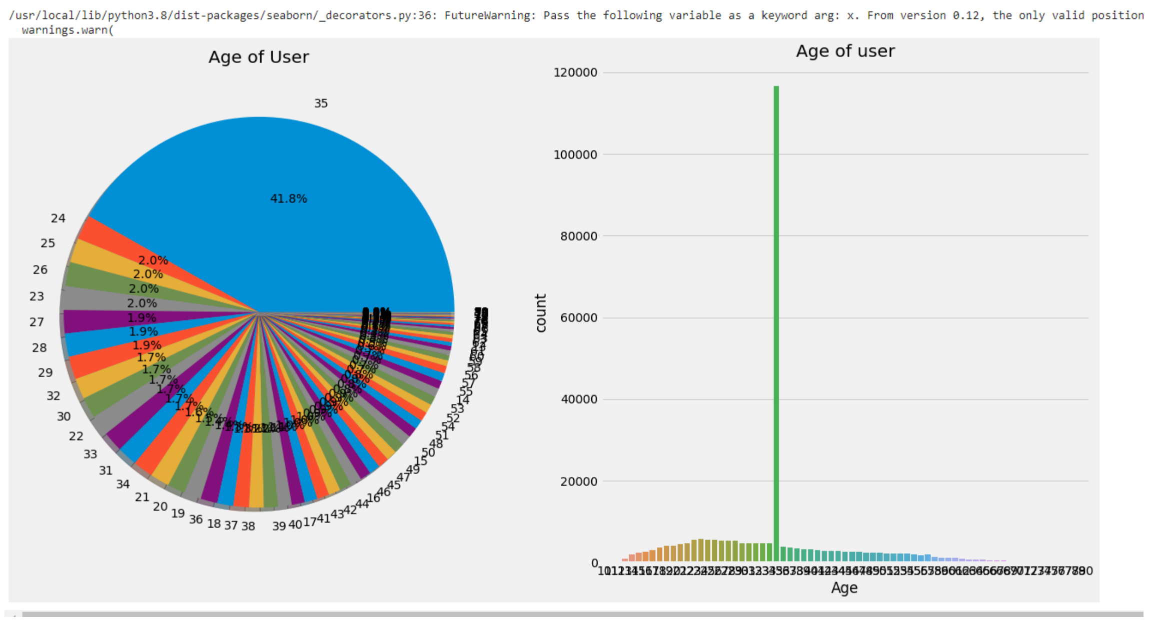
From the given distplot we can see that we have outliers in the Age column, we will treat these outliers in the coming section of outlier treatment.

Now let’s have a look at the age distribution in a range of 0 to 100 by plotting a histogram.

From the given histogram plot for age, we see that the distribution is right skewed. This information will be used further in imputing the Null values in the Age column.



We can conclude between users with age between 20-40 are highest in number

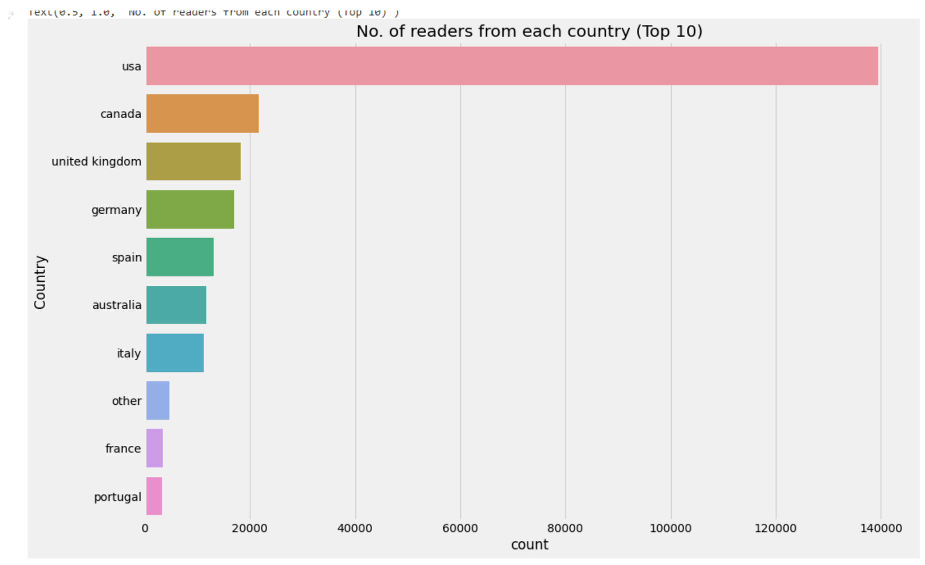


```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid position
  warnings.warn(
```

From above plots we observed that 41.9% of age 34 group read more books compared to other age groups. Also the users with the age 60 and above do not read more books.

Location: Now let’s deep dive into our location column. This column has city, state and country separated by commas. We will first segregate these into different columns and we will introduce a new column “Country” so that we can analyze on the basis of the country of different users. The following code will separate the Country from the location.

There are mis-spellings in some of the country names. We will first correct these and then plot the top 10 countries from where we have the maximum number of users. The following count plot shows the top 10 countries, here we analyzed that the maximum number of users belong to the USA.



```
Text(0.5, 1.0, 'No. of readers from each country (Top 10)')
```

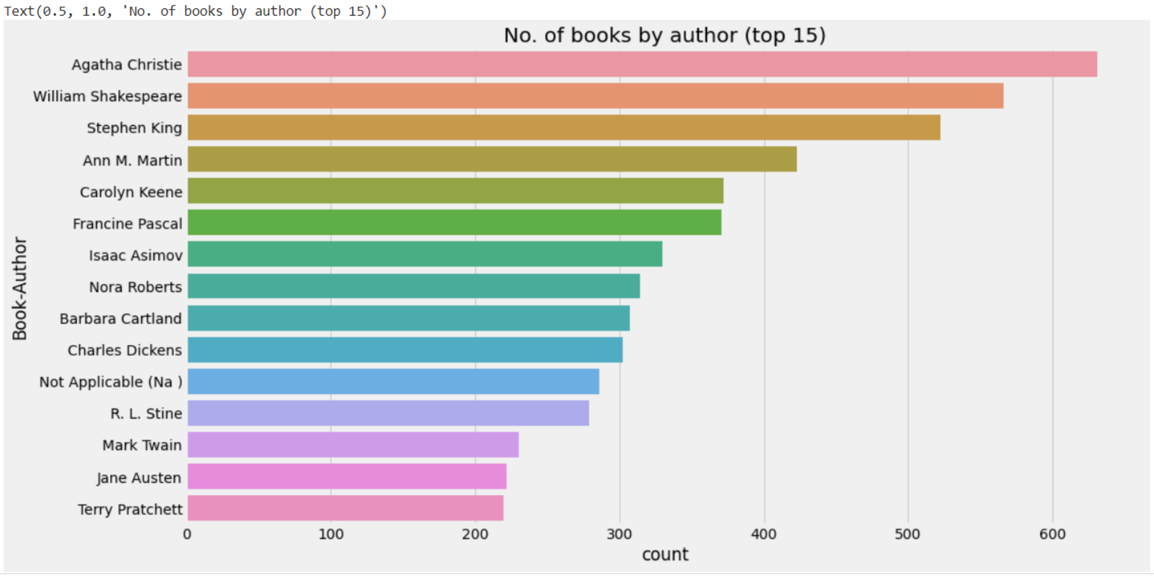
2.3.3 Book Dataset:

In the books dataset we have the following feature variables.

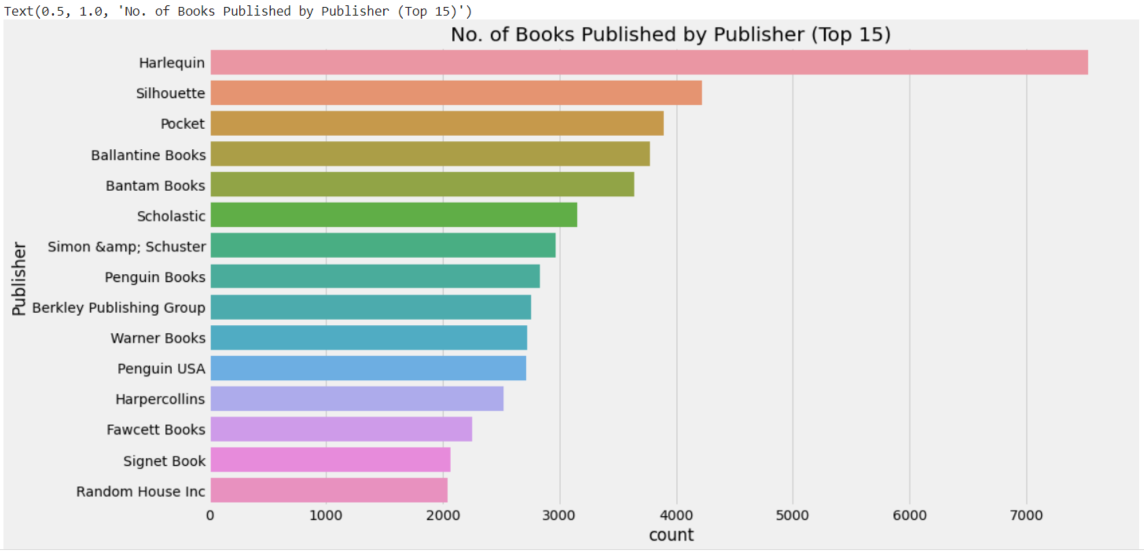
- ISBN (unique for each book)
- Book-Title
- Book-Author
- Year-Of-Publication
- Publisher
- Image-URL-S
- Image-URL-M
- Image-URL-L

From the count plot, let’s find the top 10 Book-Author and top 10 Book-Publishers. Further we find that both the plots are skewed and the maximum number of books are from top 10 Book-Authors and top 10 Book-Publishers.

```
Text(0.5, 1.0, 'No. of books by author (top 15)')
```



```
Text(0.5, 1.0, 'No. of Books Published by Publisher (Top 15)')
```



2.3.4 Ratings Dataset:

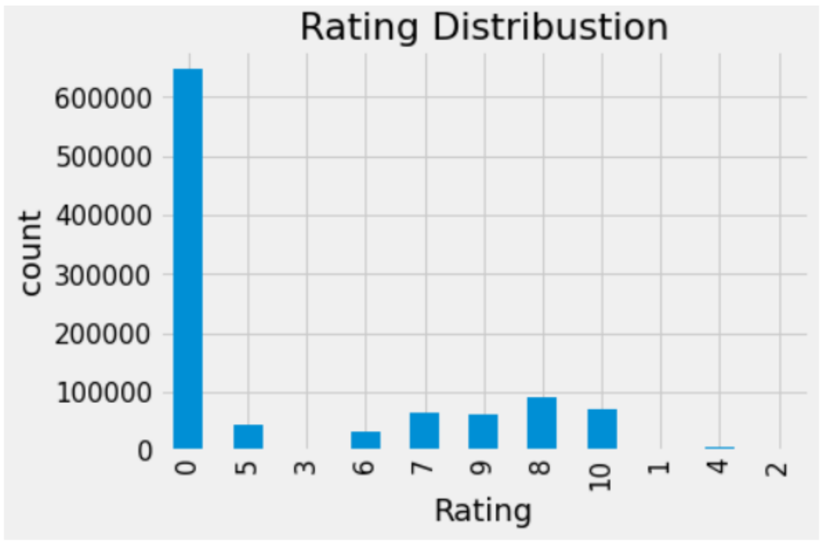
In the ratings dataset we have the following feature variables.

- User-ID
- ISBN
- Book-Rating

```
[6] ratings.head()
```

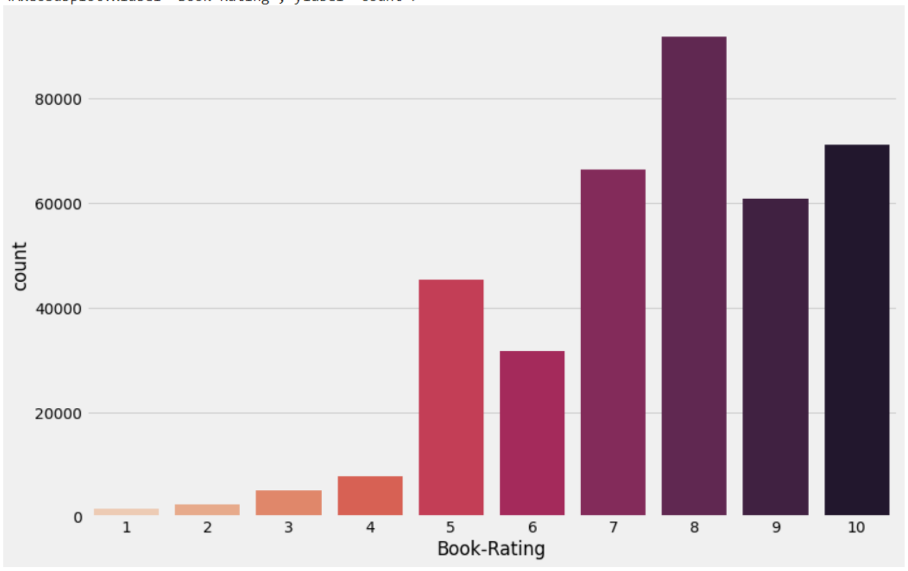
	User-ID	ISBN	Book-Rating
0	276725	034545104X	0
1	276726	0155061224	5
2	276727	0446520802	0

Ratings Distribution: Let's find the distribution of ratings frequency in our ratings dataset. From the following frequency plot, we find that most of the ratings are 0 which is implicit rating. The following code snippet will give us the frequency distribution.



The ratings are very unevenly distributed, and the vast majority of ratings are 0. As quoted in the description of the dataset — BX-Book-Ratings contains the book rating information. Ratings are either explicit, expressed on a scale from 1–10 higher values denoting higher appreciation, or implicit, expressed by 0. Hence segregating implicit and explicit ratings datasets.

```
<AxesSubplot:xlabel='Book-Rating', ylabel='count'>
```

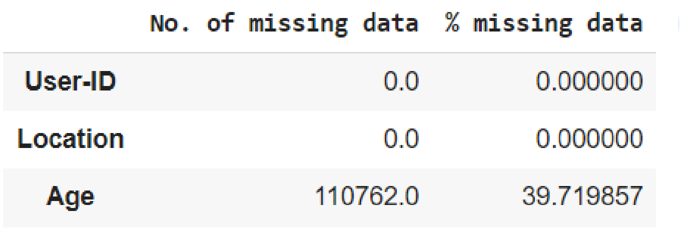


It can be observed that higher ratings are more common amongst users and rating 8 has been rated highest number of times.

2.3.5 Missing data:

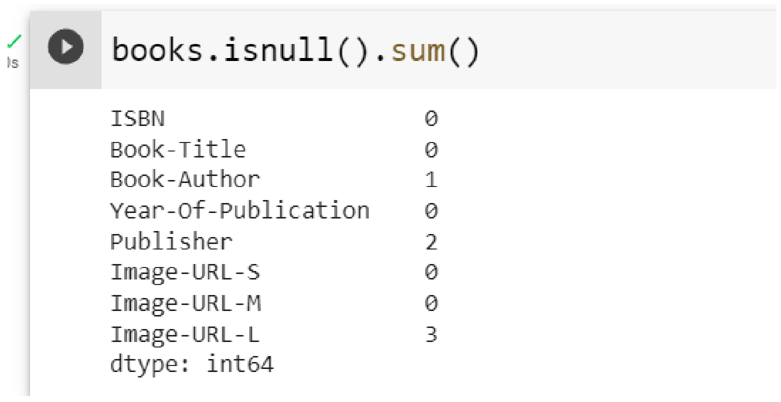
Once you have raw data, you must deal with issues such as missing data and ensure that the data is prepared for forecasting models in such a way that it is amenable to them.

There were missing values in some columns in our dataset. The following code snippet gave us the idea about missing values in different columns in users dataset.



	No. of missing data	% missing data
User-ID	0.0	0.000000
Location	0.0	0.000000
Age	110762.0	39.719857

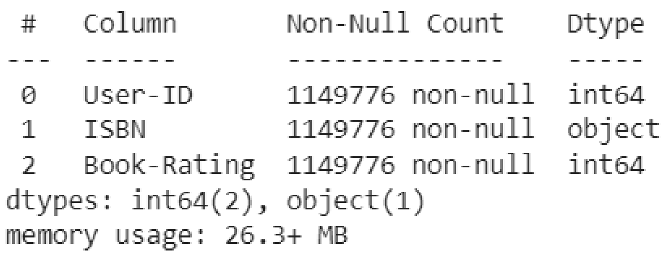
Age has around 39% missing values which we will impute in further sections.



```
books.isnull().sum()
```

```
ISBN                   0
Book-Title             0
Book-Author            1
Year-Of-Publication    0
Publisher              2
Image-URL-S            0
Image-URL-M            0
Image-URL-L            3
dtype: int64
```

We have 1 Null value in Book-Author and 2 Null values in Publisher columns of our Books data frame, these Null values we will be treating in further sections.



```
 #   Column       Non-Null Count    Dtype
---  ------       --------------    -----
 0   User-ID      1149776 non-null  int64
 1   ISBN         1149776 non-null  object
 2   Book-Rating  1149776 non-null  int64
dtypes: int64(2), object(1)
memory usage: 26.3+ MB
```

2.4 Data Pre-processing, Data Cleaning & Imputation (Handling the Categorical & Numerical Variables) –

In the data cleaning section, we will drop off the features which do not contribute towards making a good recommendation system.

Books_df: We have already seen in our EDA part that the Image-URL-S, Image-URL-M and Image-URL-L do not contribute towards our final goal. So, we will drop off these columns and also dropping these columns we will not be losing any information.

2.4.1 Handling missing values:

Values that are reported as missing may be due to a variety of factors. This lack of answers would be considered missing values. The researcher may leave the data or do data imputation to replace them. Suppose the number of cases of missing values is extremely small; then, an expert researcher may drop or omit those values from the analysis. But here in our case we had a lot of data points which were having missing values in different feature columns.

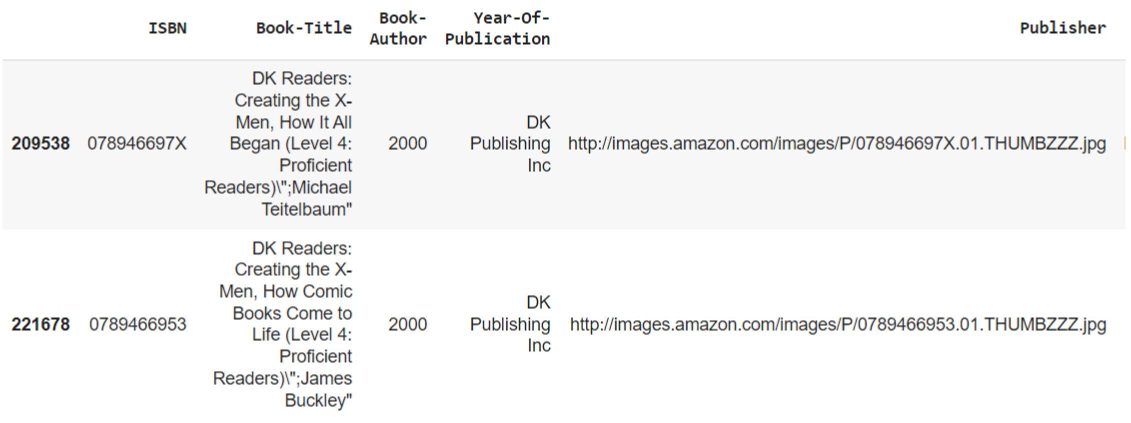
2.4.2 Imputing Values:

Since, we have missing values in some feature variables, we need to impute them.

- Year-Of-Publication
- Book-Author
- Publisher
- Age

2.4.3 Year-Of-Publication:

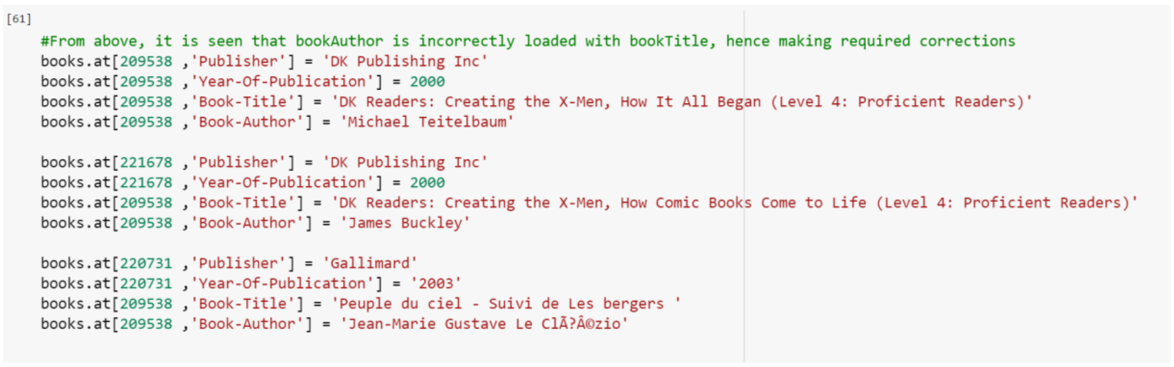
From the analysis part we get that the Year-Of-publication was wrongly mentioned for some of the rows. Diving deep into the Books_df we got to know that for these rows there was actually a column mismatch.



	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher
209538	078946697X	DK Readers: Creating the X-Men, How It All Began (Level 4: Proficient Readers)\";Michael Teitelbaum"	2000	DK Publishing Inc	http://images.amazon.com/images/P/078946697X.01.THUMBZZZ.jpg
221678	0789466953	DK Readers: Creating the X-Men, How Comic Books Come to Life (Level 4: Proficient Readers)\";James Buckley"	2000	DK Publishing Inc	http://images.amazon.com/images/P/0789466953.01.THUMBZZZ.jpg

As it can be seen from above, there are some incorrect entries in the Year-Of-Publication field. It looks like Publisher names 'DK Publishing Inc' and 'Gallimard' have been incorrectly loaded as Year-Of-Publication in the dataset due to some errors in the csv file.

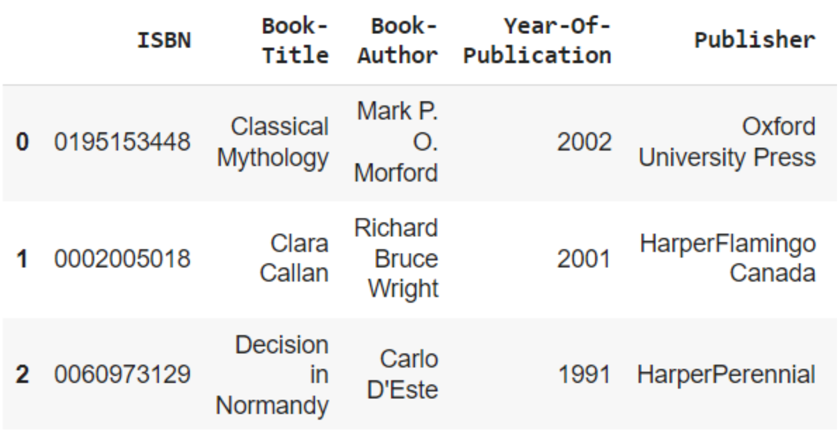
Also, since these entries were few, we could manually correct the column values for these particular rows. The following code snippet shows the imputation.



```
[61]
    #From above, it is seen that bookAuthor is incorrectly loaded with bookTitle, hence making required corrections
    books.at[209538 ,'Publisher'] = 'DK Publishing Inc'
    books.at[209538 ,'Year-Of-Publication'] = 2000
    books.at[209538 ,'Book-Title'] = 'DK Readers: Creating the X-Men, How It All Began (Level 4: Proficient Readers)'
    books.at[209538 ,'Book-Author'] = 'Michael Teitelbaum'

    books.at[221678 ,'Publisher'] = 'DK Publishing Inc'
    books.at[221678 ,'Year-Of-Publication'] = 2000
    books.at[209538 ,'Book-Title'] = 'DK Readers: Creating the X-Men, How Comic Books Come to Life (Level 4: Proficient Readers)'
    books.at[209538 ,'Book-Author'] = 'James Buckley'

    books.at[220731 ,'Publisher'] = 'Gallimard'
    books.at[220731 ,'Year-Of-Publication'] = '2003'
    books.at[209538 ,'Book-Title'] = 'Peuple du ciel - Suivi de Les bergers '
    books.at[209538 ,'Book-Author'] = 'Jean-Marie Gustave Le ClÃ?Â©zio'
```



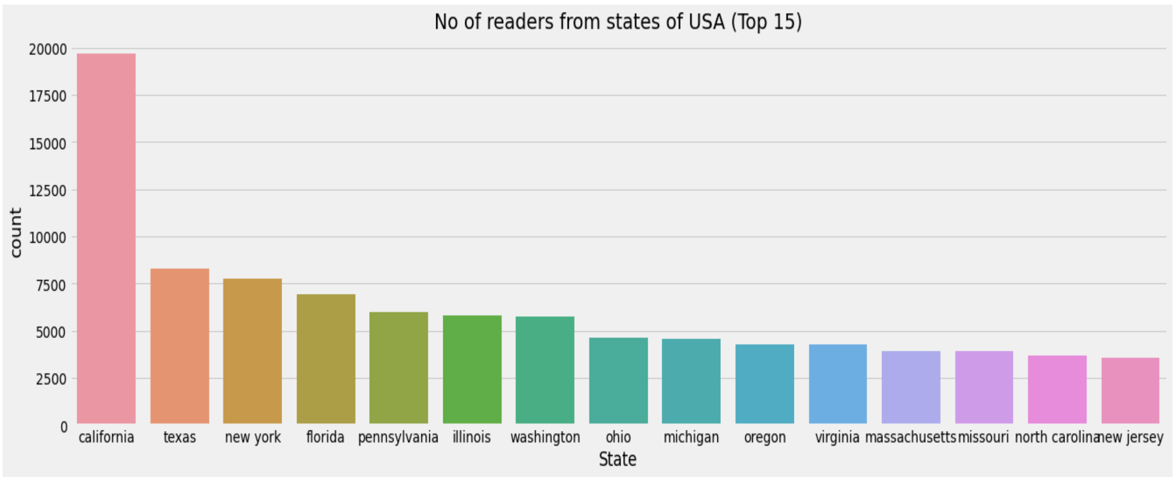
	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial

2.5 Analyse the Data –

Once the pre-processing is done, we are good to go with our actual analysis where we write lines of codes and logics to prepare our data as per the defined use cases.

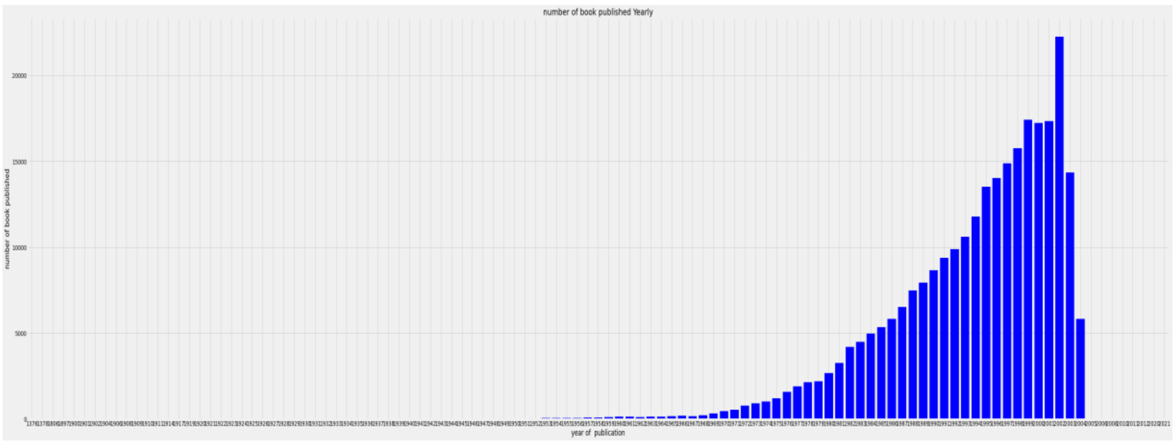
2.5.1 Top 15 readers from states of USA:

- Splitting Location column and analyzing State of USA.
- Most active readers are from California.



2.5.2 Number of books published by year count:

So, we can observe that publication years are somewhat between 1950 - 2005 here.



Chapter 3

Technology Used

3.1. Python

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

3.2. Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

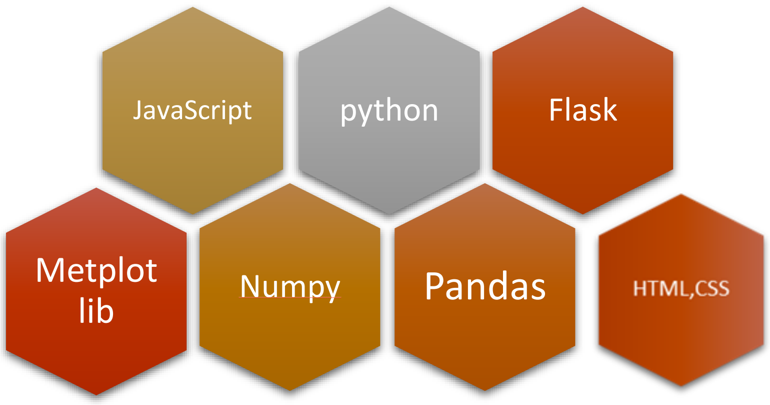


Fig 3. Technology used.

3.3. Pandas

Pandas is a Python library for data analysis. **Started by Wes McKinney in 2008** out of a need for a powerful and flexible quantitative analysis tool, pandas has grown into one of the most popular Python libraries.

3.4. NumPy

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. **NumPy was created in 2005 by Travis Oliphant**. It is an open-source project, and you can use it freely.

3.5. Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

3.6 Flask

Flask is a web framework, it's a Python module that lets you develop web applications easily. It's has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features.

It does have many cool features like URL routing, template engine. It is a WSGI web app framework.

Chapter 4

4.1 Popularity Based Approach:

It is a type of recommendation system which works on the principle of popularity and or anything which is in trend. These systems check about the books which are in trend or are most popular among the users and directly recommend them.

For example, if a product is often purchased by most people, then the system will get to know that that product is most popular so for every new user who just signed it, the system will recommend that product to that user also and chances become high that the new user will also purchase that.

4.1.1 Why Is this model relevant?

The answer to this is the **Cold-Start Problem.** Cold start is a potential problem in computer-based information systems which involves a degree of automated data modelling. Specifically, it concerns the issue that the system cannot draw any inferences for users or items about which it has not yet gathered sufficient information. The cold start problem is a well-known and well researched problem for recommender systems. Recommender systems form a specific type of information filtering (IF) technique that attempts to present information items (e-commerce, films, music, books, news, images, web pages) that are likely of interest to the user. Typically, a recommender system compares the user's profile to some reference characteristics. These characteristics may be related to item characteristics (content-based filtering) or the user's social environment and past behavior (collaborative filtering). Depending on the system, the user can be associated with various kinds of interactions: ratings, bookmarks, purchases, likes, number of page visits etc.

There are three cases of cold start:

- **New community:** refers to the start-up of the recommender, when, although a catalogue of items might exist, almost no users are present, and the lack of user interaction makes it very hard to provide reliable recommendations

- **New item:** a new item is added to the system, it might have some content information, but no interactions are present

- **New user:** a new user registers and has not provided any interaction yet, therefore it is not possible to provide personalized recommendations.

A popularity-based model does not suffer from cold start problems which means on day 1 of the business also it can recommend products on various different filters. There is no need for the user’s historical data. The popularity index used for our books dataset was **weighted rating**.

$$WR = [(v * R)/ (v + m)] + [(m * c)/(v + m)]$$

where,

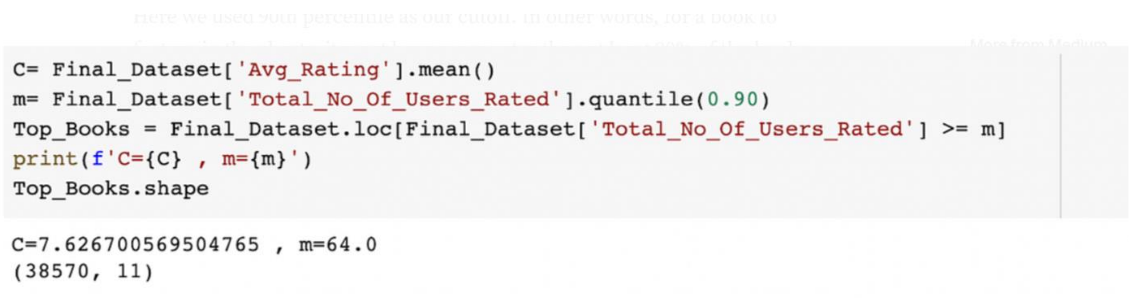
v is the number of votes for the books;

m is the minimum votes required to be listed in the chart.

R is the average rating of the book; and

C is the mean vote across the whole report.

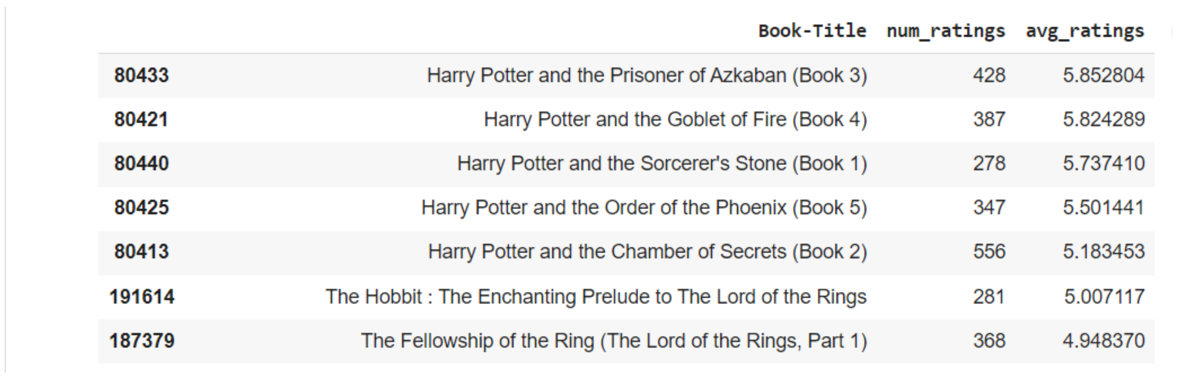
The function used to calculate **C** and **m** is given below.



```
C= Final_Dataset['Avg_Rating'].mean()
m= Final_Dataset['Total_No_Of_Users_Rated'].quantile(0.90)
Top_Books = Final_Dataset.loc[Final_Dataset['Total_No_Of_Users_Rated'] >= m]
print(f'C={C} , m={m}')
Top_Books.shape
```

```
C=7.626700569504765 , m=64.0
(38570, 11)
```

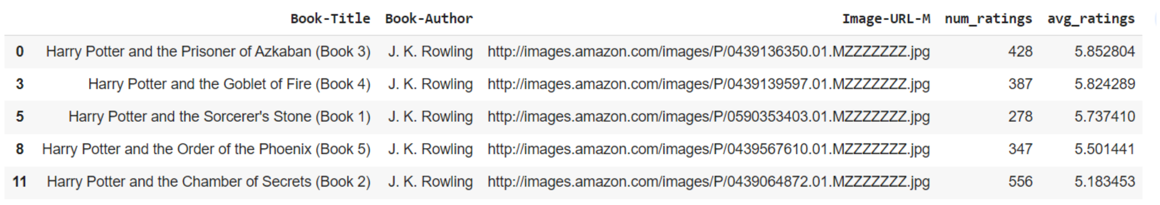
Using this popularity metric we can calculate the top books that could be recommended to a user.



	Book-Title	num_ratings	avg_ratings
80433	Harry Potter and the Prisoner of Azkaban (Book 3)	428	5.852804
80421	Harry Potter and the Goblet of Fire (Book 4)	387	5.824289
80440	Harry Potter and the Sorcerer's Stone (Book 1)	278	5.737410
80425	Harry Potter and the Order of the Phoenix (Book 5)	347	5.501441
80413	Harry Potter and the Chamber of Secrets (Book 2)	556	5.183453
191614	The Hobbit : The Enchanting Prelude to The Lord of the Rings	281	5.007117
187379	The Fellowship of the Ring (The Lord of the Rings, Part 1)	368	4.948370

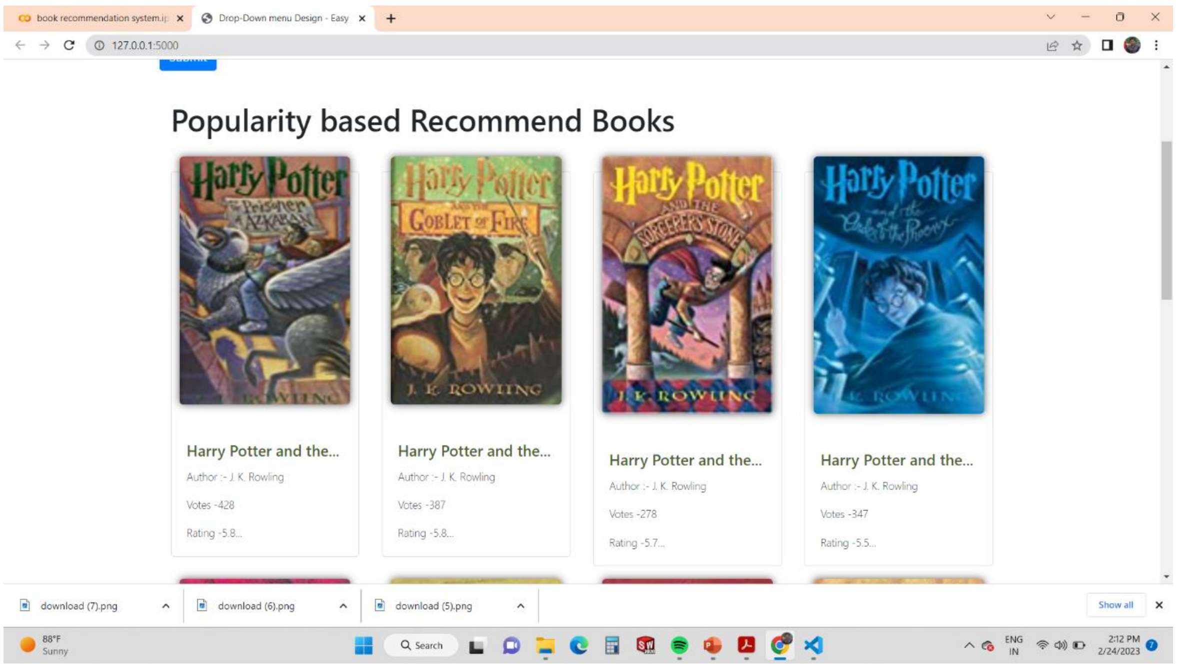
Hence it was observed that the **Harry Potter series** was the most popular book series. Other than that, books like **To Kill A Mockingbird**, **The Lord of the Rings** and **Dune** were among the top books.

The **Popularity Based Recommender** provides a general chart of recommended movies to all the users. They **are not sensitive to the interests and tastes of a particular user**. It is not personalized and the system would recommend the same sort of products/movies which are solely based upon popularity to every other user. Hence, we need to try and work out other kinds of recommender systems.



	Book-Title	Book-Author	Image-URL-M	num_ratings	avg_ratings
0	Harry Potter and the Prisoner of Azkaban (Book 3)	J. K. Rowling	http://images.amazon.com/images/P/0439136350.01.MZZZZZZZ.jpg	428	5.852804
3	Harry Potter and the Goblet of Fire (Book 4)	J. K. Rowling	http://images.amazon.com/images/P/0439139597.01.MZZZZZZZ.jpg	387	5.824289
5	Harry Potter and the Sorcerer's Stone (Book 1)	J. K. Rowling	http://images.amazon.com/images/P/0590353403.01.MZZZZZZZ.jpg	278	5.737410
8	Harry Potter and the Order of the Phoenix (Book 5)	J. K. Rowling	http://images.amazon.com/images/P/0439567610.01.MZZZZZZZ.jpg	347	5.501441
11	Harry Potter and the Chamber of Secrets (Book 2)	J. K. Rowling	http://images.amazon.com/images/P/0439064872.01.MZZZZZZZ.jpg	556	5.183453

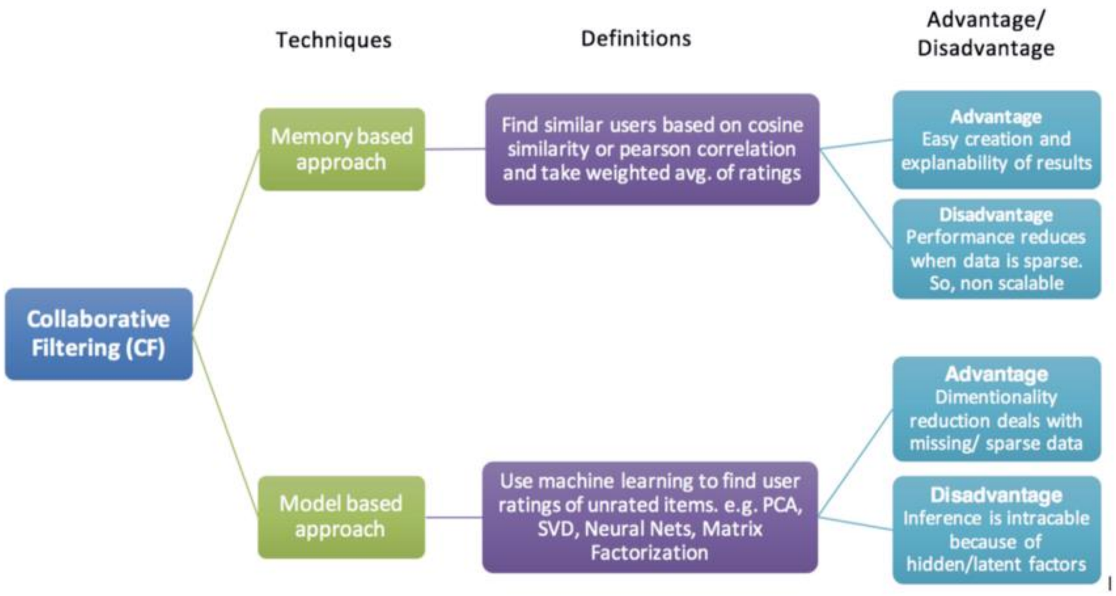
Webpage Screenshot:



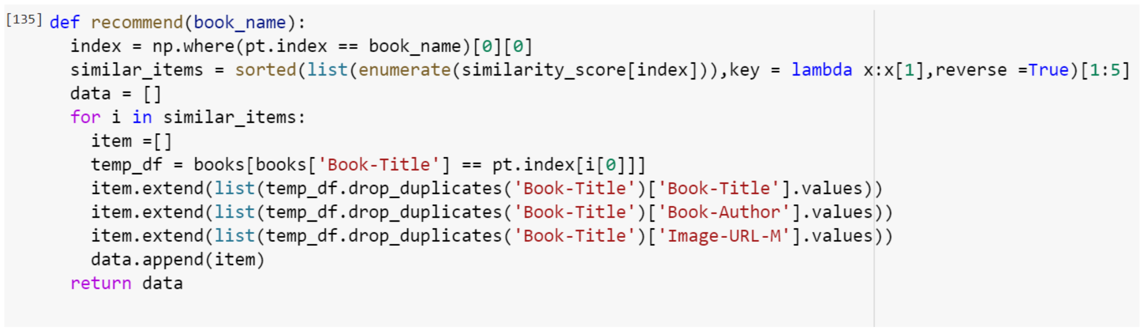
4.2 Collaborative Filtering:

It is considered to be one of the very smart recommender systems that work on the similarity between different users and items that are widely used as an e-commerce website and also online movie websites. It checks about the taste of similar users and makes recommendations.

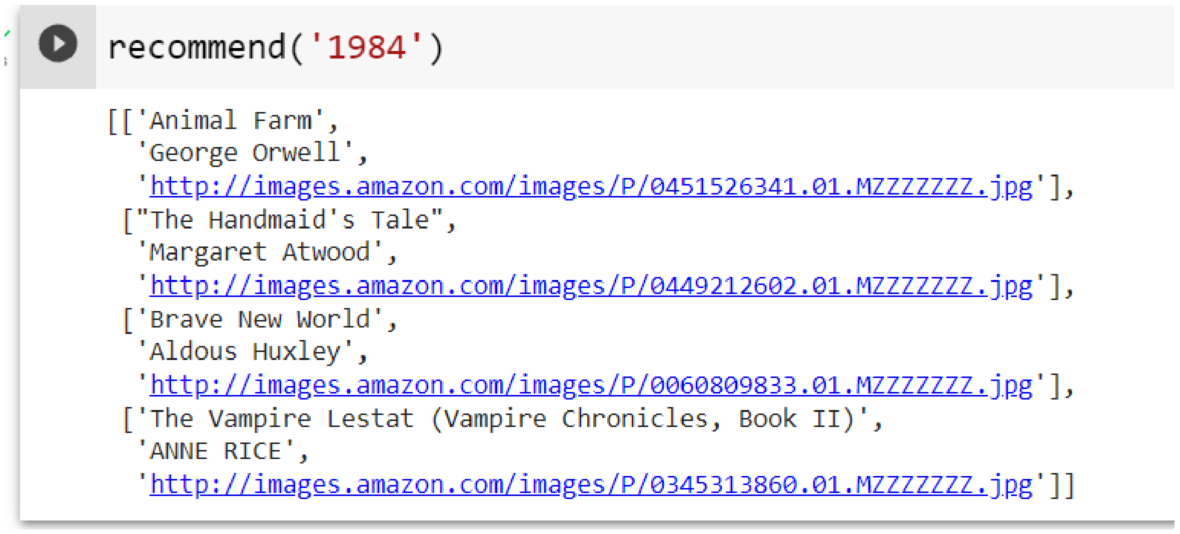
The similarity is not restricted to the taste of the user, moreover there can be consideration of similarity between different items also. The system will give more efficient recommendations if we have a large volume of information about users and items. There are various types of collaborative filtering techniques as mentioned in the diagram given below.


Techniques
Definitions
Advantage/ Disadvantage
Collaborative Filtering (CF)
Memory based approach
Find similar users based on cosine similarity or pearson correlation and take weighted avg. of ratings
Advantage
Easy creation and explanability of results
Disadvantage
Performance reduces when data is sparse. So, non scalable
Model based approach
Use machine learning to find user ratings of unrated items. e.g. PCA, SVD, Neural Nets, Matrix Factorization
Advantage
Dimentionality reduction deals with missing/ sparse data
Disadvantage
Inference is intracable because of hidden/latent factors

This is function are used for collaborative function.



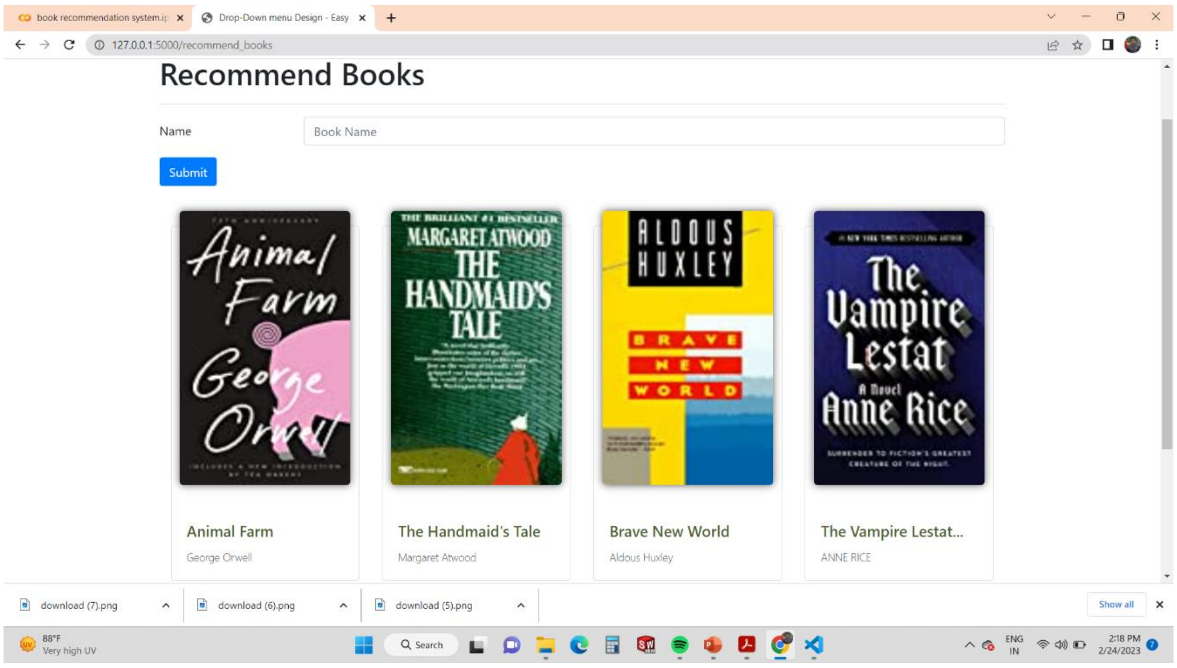
```
[135] def recommend(book_name):
        index = np.where(pt.index == book_name)[0][0]
        similar_items = sorted(list(enumerate(similarity_score[index])),key = lambda x:x[1],reverse =True)[1:5]
        data = []
        for i in similar_items:
          item =[]
          temp_df = books[books['Book-Title'] == pt.index[i[0]]]
          item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-Title'].values))
          item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-Author'].values))
          item.extend(list(temp_df.drop_duplicates('Book-Title')['Image-URL-M'].values))
          data.append(item)
        return data
```



```
recommend('1984')
```

```
[['Animal Farm',
  'George Orwell',
  'http://images.amazon.com/images/P/0451526341.01.MZZZZZZZ.jpg'],
 ["The Handmaid's Tale",
  'Margaret Atwood',
  'http://images.amazon.com/images/P/0449212602.01.MZZZZZZZ.jpg'],
 ['Brave New World',
  'Aldous Huxley',
  'http://images.amazon.com/images/P/0060809833.01.MZZZZZZZ.jpg'],
 ['The Vampire Lestat (Vampire Chronicles, Book II)',
  'ANNE RICE',
  'http://images.amazon.com/images/P/0345313860.01.MZZZZZZZ.jpg']]
```

Webpage Screenshot



Recommend Books

Name

Book Name

Submit

Animal Farm

George Orwell

The Handmaid's Tale

Margaret Atwood

Brave New World

Aldous Huxley

The Vampire Lestat...

ANNE RICE

Chapter 5

5.1 Challenges Faced:

- ❖ Reading the dataset and understanding the meaning of some columns.
- ❖ For answering some of the questions we had to understand
- ❖ Book Recommendation System model that how they work.
- ❖ Handling Nan values, null values, and duplicates.
- ❖ Designing multiple visualizations to summarize the information in the dataset and successfully communicate the results and trends to the reader.

5.2 Problems Faced:

- **Handling of sparsity** was a major challenge as well since the user interactions were not present for the majority of the books.
- Understanding the metric for evaluation was a challenge as well.
- Since the data consisted of text data, **data cleaning** was a major challenge in features like **Locatio**n etc..
- **Decision making on missing value imputations and outlier treatment** was quite challenging as well.

5.3 Scope of Improvement:

- Given more information regarding the books dataset, namely features like Genre, Description etc., we could implement a content-filtering based recommendation system and compare the results with the existing collaborative-filtering based system.
- We would like to explore various clustering approaches for clustering the users based on Age, Location etc., and then implement voting algorithms to recommend items to the user depending on the cluster into which it belongs.

5.4 Conclusion:

- In EDA, the Top-10 most rated books were essentially **novels**. Books like **The Lovely Bone** and **The Secret Life of Bees** were very well perceived.

- Majority of the readers were of the **age bracket 20–35** and most of them came from North American and European countries namely **USA, Canada, UK, Germany and Spain**.

- If we look at the ratings distribution, **most of the books have high rating**s with maximum books being rated 8. Ratings below 5 are few in number.

- Author with the most books was **Agatha Christie, William Shakespeare and Stephen King**.

- For modelling, it was observed that for **model based** collaborative filtering SVD technique worked way better than NMF with lower Mean Absolute Error (MAE) .