VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ADVANCED SCIENCE - (SAS)

Winter Semester - 2021-2022

ONLINE BOOK RECOMMENDATION SYSTEM

PROJECT FINAL REVIEW - 1

Submitted fulfillment for the J-component of CSC5007

EXPLORATORY DATA ANALYSIS

CAL COURSE
In

MSC. DATASCIENCE

BY

JOTHIKA R - 21MDT0052
LOSHMIN G K - 21MDT0027
NANDHINI R - 21MDT0103
KISHORKUMAR K R - 21MDT0088
SANTHOSH KUMAR S - 21MDT0132

Under the guidance of

Prof. PALLAVI MISHRA

ABSTRACT:

Today the amount of information in the internet growth very rapidly and people need some instruments to find and access appropriate information. But since data is huge and complex it is difficult to get useful information from it recommendation system are effective software techniques to overcome this problem. Recommendation system is one of most popular applications of artificial intelligence which attracts many researches all over the global and help to navigate quickly and receive necessary information. There are many machine learning techniques which can be used to realize the recommendation system. This paper proposes a quick and intuitive book recommendation systems that helps the readers to find appropriate book to read next and predict buyers interest and recommend books to them accordingly. The work will help the researchers in exploring new dimension for recommendation technology in general and book recommendation in particular.

INTRODUCTION:

Growth of the internet and advancement in computing technologies has touched every aspect of human being. Nowadays computers are being implemented in almost every domain of life to replicate the manual system and to reduce the workload of humans. Recommender are a type of intelligent systemswhich are implemented to replicate human experts. They have been implemented in various real-life applications to assist people in decision making. Book recommenders have been designed considering the changing pace of time. They are implemented in libraries for their maximum utilization. By mining the borrowing records of libraries, it suggests the librarians the books to be purchased. On the basisof browsing history of a library user a recommender suggests him the books which he may like or which are related to his area of study if he is a student or a teacher. Similarly, a book recommender applied in e-commerce domain suggest the merchantsabout the different books and help him to manage his inventory. It also helps the user in purchasing the most appropriate book for him considering various criteria like his preferences, cost and other features of Book.

Machine learning (ML) is one of the fascinating and fastest growing field of this era. ML techniques are being implemented in almost every domain of computing. They have been implemented to improve the predictive and recommendation accuracy of the system. These techniques have been utilized for extraction and classification of book features. Human sentiments and their choices have also been extracted using these. After finding sentiments, choices, browsing history, and features of books similarity is calculated and recommendations are generated. This paper is a study on machine learning techniques which have been implemented on book recommenders. This paper starts with introduction in section 1. In section 2, machine learning techniques have been discussed which is followed by literature survey.

PROBLEM STATEMENT:

A recommendation system will help users who do not have enough individual knowledge to peruse through the different types of options offered by a website. It will provide the users with information to assist them to make a decision as to which items to purchase. The proposed work alters from the existing recommendation systems since the existing ones only consider one technique to recommend items to the users. They do not recommend items using two or more techniques and are not a Hybrid Recommendation System. The proposed system uses combination of Collaborative Filtering and Association Rule Mining. Collaborative Filtering is used to predict the ratings of a particular item by calculating ratings given to similar items or ratings given by similar users. Association Rule Mining is used to extract different patterns and correlations between different items. Thus, the usage of both these techniques can help to control the data sparsely problem and cold start problem in recommendation system.

LITERATURE SURVEY:

Title and Year	Algorithm Technique	Advantages	Disadvantages
TITLE Book Recommendation System Based on Collaborative Filtering and Association Rule Mining.[1] **YEAR :** 2018	• Classification Technique. • Association rule mining. • Collaborative Filtering	They considered user based collaborative filtering which helped in getting the books of good quality. Its helped in filtering the transaction to find stronger ecommendation and classification technique.	It does not have a exact accuracy of recommondation book with the past mining techniques.
TITLE : Book Recommendation System with Tensor Flow.[2] **YEAR** : 2021	• Collaborative Filtering • Deep learning • Tenserflow	Tensorflow techniques to make a recommendation system stable and recommend the relevant kind of books to the user in the efficient and simple manner. The accuracy and effectiveness of this project from obtained results and visualization models.	In this techniques they doesn't focus on the security system. They use the old technique for the recommendation need to add a new techniques for better accuracy.
TITLE : Book Recommendation for eLearning Using Collaborative Filtering and Sequential Pattern Mining[3]. **YEAR : 2020**	• Sequential pattern mining • PrefixSpan • Collaborative Filtering	Manhattan gives a lesser error rate when only collaborative filtering is applied. PrefixSpan on the prediction matrix, improves the recommendation accuracy and overcomes	It can be extended to cross domain with context-awareness to provide better recommendation accuracy and overcome cold-start. It also has a sparsity problem in

		the new user problem.	recommendation.
TITLE Online Book Recommendation System using Collaborative Filtering (With Jaccard Similarity)[4]. **YEAR** : 2019	• Similarity index • Filtering techniques • Jaccard Similarity	JS uses the number of common users as used by most existing algorithms, which gives a more accurate result.	This project build with rating recommendation so, It might arise from a book that a user has given low rating to, in which case a book might be recommended from a genre that the user dislikes. This recommendation system does not solve the trust issue.
TITLE Book Recommendation Using Machine Learning Methods Based on Library Loan Records and Bibliographic Information.[5] **YEAR:** 2020	• Support Vector Machine (SVM) • Random Forest. • Adaboost.	This recommendation system works in offline mode, so it does not have any performance problem.	They have a problem with the effectiveness of incorporating into recommendation methods information.
TITLE An Online Book Recommendation System Based on Web Service.[6] **YEAR : 2018**	• Web service • Long tail effect • Web Craw	Recommendation pages contain all the essential and expanding book information for readers to refer to. Readers can recommend a book on these pages, and recommendation data will be analysed by the recommendation system to make scientific purchasing decision.	The precise effect still should be verified through long-term trial.

TITLE Book Recommendation System.[7] **YEAR: 2019**	• KNN • Collaborative Filtering	Individuals' reading habits are improved by recommending books and this will improve their vocabulary, knowledge and skill.	It takes some time to recommend the books using Collaborative Filtering method since the dataset is large.
TITLE Development of an Open-Source Automated Library System with Book Recommendation System for Small Libraries[8]. YEAR: 2020	• Support Vector Machine • Artificial Intelligence	They were conducted and compared against three features: title similarity, bibliographies similarity, combination of title and bibliographies. In our research, the database are collect from School library in chiangmai with consist of material book around 3000-10000 books	Algorithm is recommend for small organizations of automated library systems with less than 10000 books and not too much loan record of users such as libraries of schools or small organizations.
TITLE Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining[9]. YEAR: 2016	• Association rule • Collaborative filtering • Content based Filtering	This paper presents a new approach for recommending books to the buyers. This system combines the features of content filtering, collaborative filtering and association rule mining to produce efficient and effective recommendations.	This system does not have performance problem since it built the recommendations offline. It does not recommendations will be generated automatically
TITLE Machine Learning Algorithm for Recommender System – A Comparative Analysis [10]. YEAR: 2017	• Recommender System, • Classifier, • Content Based • Collaborative Based • Clusters.	This comprehensive analysis depicts the strength and the weakness of each one of them in different versions of the MovieLens dataset.	There is a need of high accuracy. Such systems still have space for improvement.

Overview of the proposed system:

INTRODUCTION:

Problem statement:

Students find it difficult searching for books based on subjects, especially when it comes to answering entrance level exams and also during their academic exams. Searching through various sites onto the internet and determining which book is appropriate for buying or referring to, is atedious job for students. This paper provides solution to this problem by providing a subject-based books recommendation platform using deep learning algorithms.

MODULES OF PROPOSED SYSTEM :

The online book recommendation system involves various techniques for providing effective suggestion for the buyers. The association mining, collaborative filtering and content filtering are the three widely employed methods for strong impact using search engines. The content based filtering system is one in which the recommendation to the buyers are provided based on the items they have searched for. The items are generally in the form of text, comprising e-mail and web pages. This method analyse the similarities between the items to bring out the best recommendation.

The collaborative filtering involves the analysis of the opinions in which the recommendation is provided based on the ratings provided by the users. The quality of the item cannot be analysed in the content based filtering. But the collaborative filtering can expose the quality of the item. The collaborative filtering is employed in two ways namely, the user based collaborative filtering and item based collaborative filtering.

The next process to be performed is association rule mining in which association and correlation relationship is mined for the best outcome. The exemplar for the association rule mining would be market basket in which the set of data are analysed to obtain the buying pattern of the user. The two measures namely, support and confidence is used for analysis.

Collecting the dataset:

Collecting Dataset for the book recemmondation system from kaggle. This dataset will taken in e-commerce Amazon website dataset. Its contains user data, book data and rating book data.

Users:

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.

Building the Machine learning algorithm :

Collaborative Filtering :

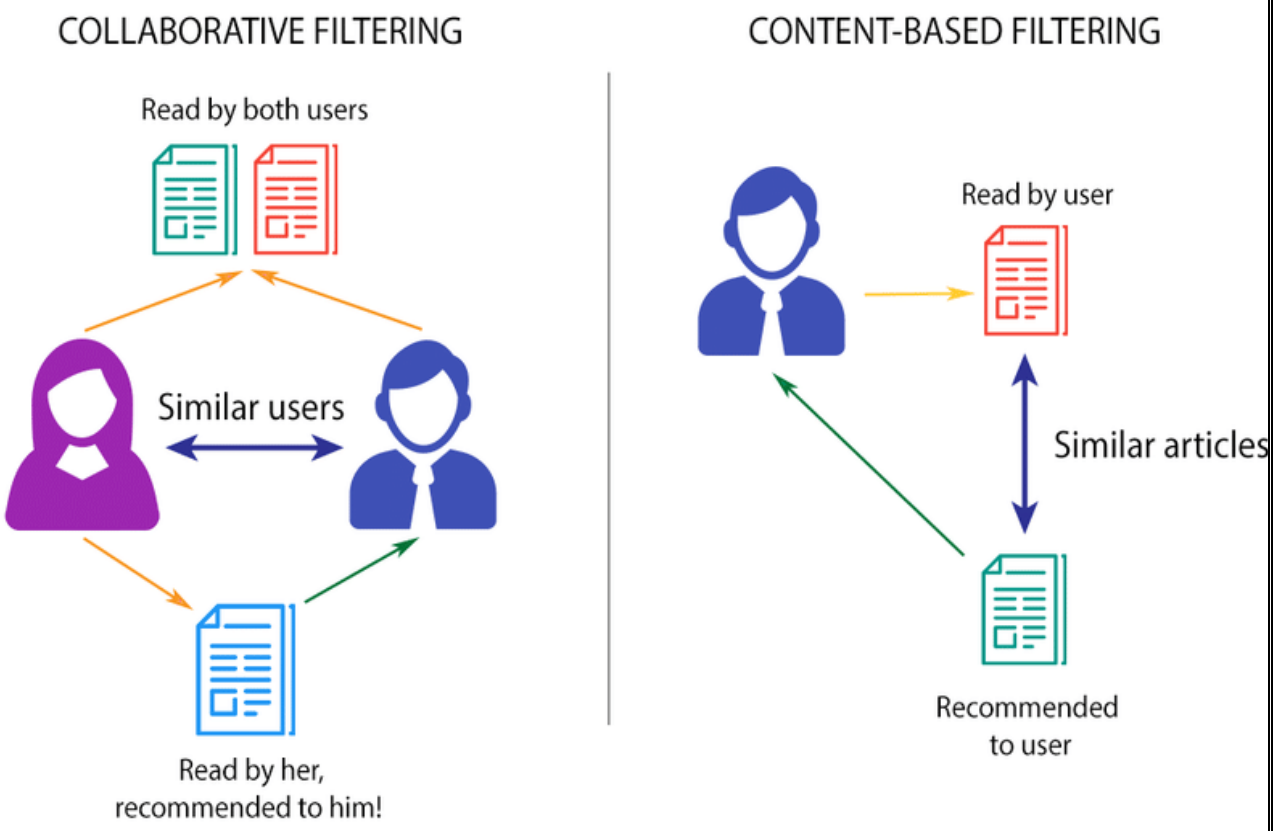
This Online book selling websites helps to buy the books online with Recommendation system which is one of the stronger tools to increase profit and retaining buyer. The book recommendation system must recommend books that are of buyer's interest. Recommendation systems are widely used to recommend products to the end users that are most appropriate. This system uses features of collaborative filtering to produce efficient and effective recommendations. Collaborative recommendation is probably the most familiar, most widely implemented and most mature of the technologies. Collaborative recommender systems aggregate ratings of objects, recognize commonalities between users on the basis of their ratings, and generate new recommendations.

K-Nearest Neighbors algorithm:

Recommendation Systems and learn how to build a book Recommendation System using collaborative filtering by implementing the K-Nearest Neighbors algorithm. We will also predict the rating of the given movie based on its neighbors and compare it with the actual rating.

Content-based filtering

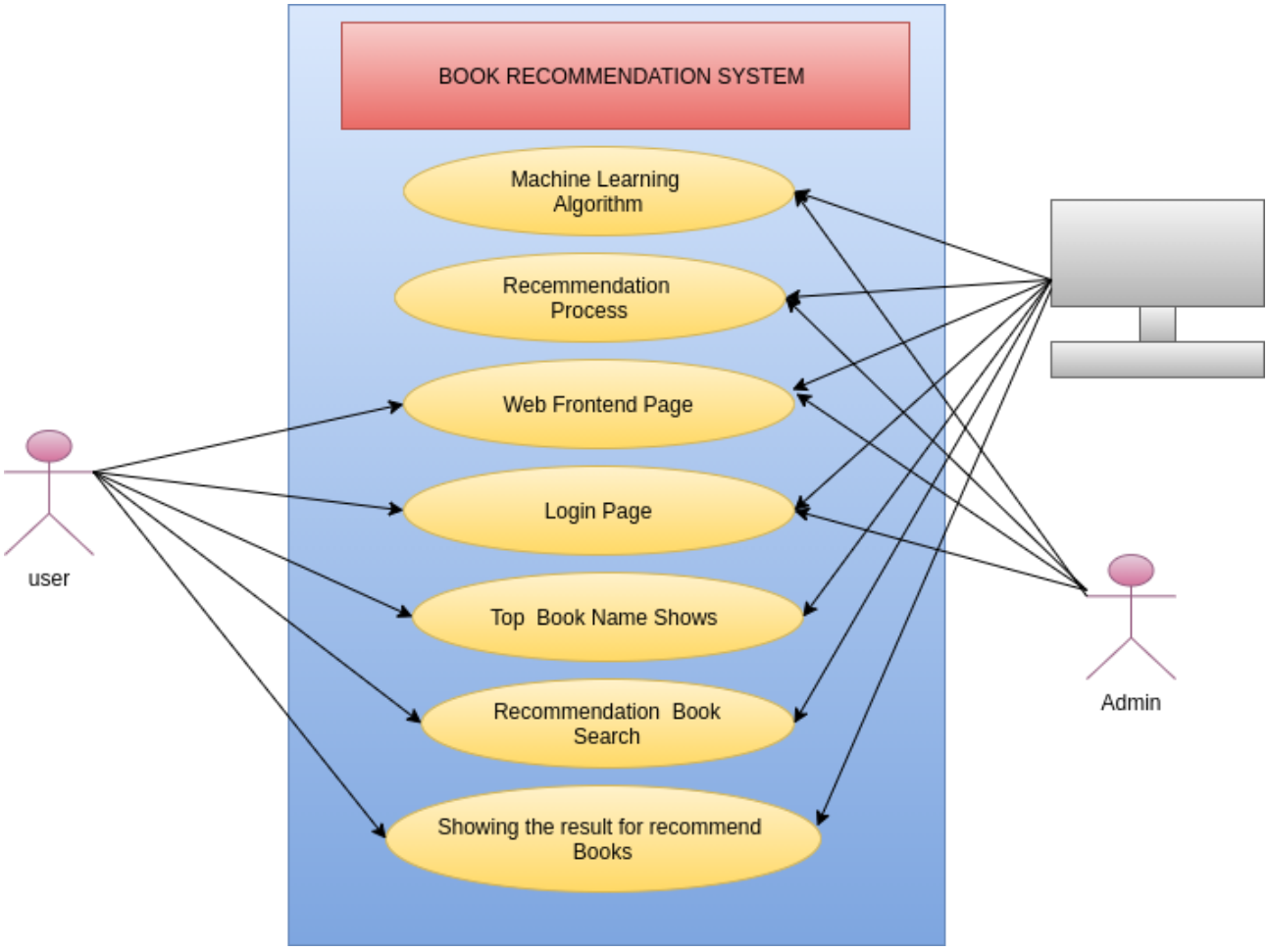
These filtering methods are based on the description of an item and a profile of the user's preferred choices. In a content-based recommendation system, keywords are used to describe the items, besides, a user profile is built to state the type of item this user likes. In other words, the algorithms try to recommend products that are similar to the ones that a user has liked in the past.



Integrating Machine Learning into Web Application with Flask:

By using python flask package . We develop a website through py charm Application. In the we use the HTML, CSS , BOOSTRAP and Java script to build the user interface (frontend) as a website. Integrate the algorithm collaborative filtering and knn using Pickle to we export the data and import the flask package then develop a website for the recommendation system.

UML DAIGRAM FOR BOOK RECOMMENDATION SYSTEM :



PROPOSED SYSTEM AND ANALYSIS:

In this project, we propose a book recommendation system. The project is divided into two segments such that one segment takes subject name as input from the user and processes it. After processing, we use cosine similarity measure to find the similar books related to the subject from sites like Amazon and Flip kart . We take the datasets from these sites and convert the datasets into trained data and use these for finding the similarities. In the second segment we create a recommendation system using (KNN) and collaboration technique, where we list the books based on the book covers given as the input. The book

cover dataset is obtained through Kaggle site. Once the user clicks on the required book, the system recommends the books similar to the book covers

The project is divided into two segment:

1. Recommendation using text processing and similarity measures.
2. Recommendation using image classification.

3.1 Working :

3.1.1 Recommendation using text processing and similarity measures:

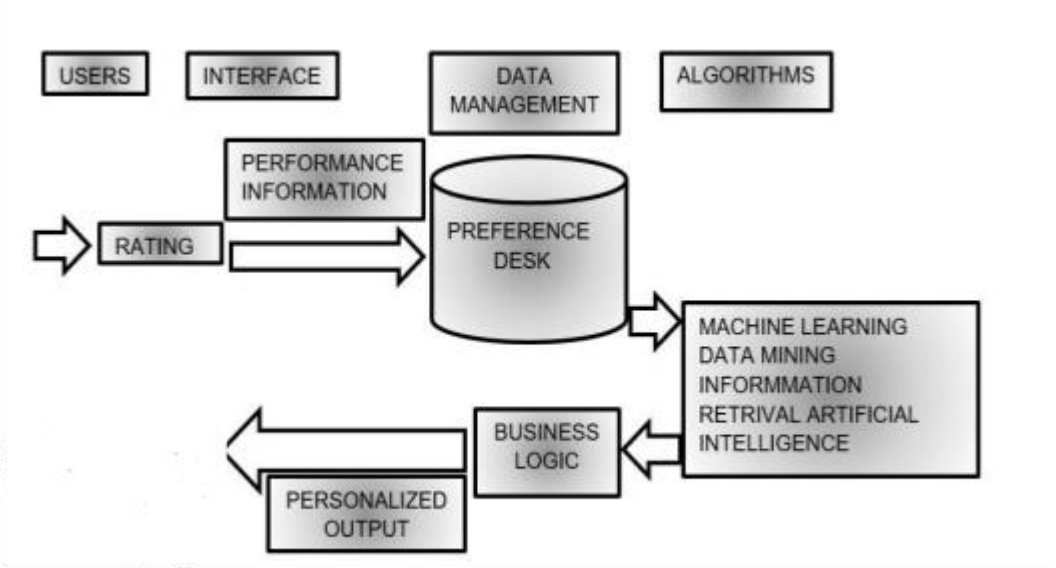
I) Input In this segment, we first take the input as the subject name and process it accordingly.

The input can be of two types:

1. Name of the subject (As specified normally) – The output from this module directly goes to the classification module without any further processing.
2. Input can be long terms and reference links – In this case, is undergoes text pre-processing for identifying only the useful and meaningful terms.

II) Datasets:

The required datasets are web scrapped from websites like Amazon and Flip kart. The datasets contains the book name, author, ratings, customers rated and the price. The datasets of amazon and flip kart are processed separately and converted to csv files. Depending on the user input the books are searched with similarity measures in the particular dataset of Amazon or Flip kart.



III) Working :

The input as subject name is concatenated with the dataset of user preference and based on the category selected by the user. The concatenated data is stored as (input, book dataset). Then the pairs of input and books in dataset are formed as (input, book). The book names in the dataset are retrieved and normalization of data is performed. Normalization of data includes tokenization, stopword elimination, stemming and POS tagging. This normalized data is fitted into TFidfvectorizer and a feature matrix is created. The pairs formed are mapped with the index of the pairs and feature matrix of the document. Then the cosine similarity of the pairs is calculated.

IV) Output:

The output of the cosine similarity function is the similarity scores between the pairs. The pairs having the highest similarity scores are listed and the books corresponding to the index pairs are retrieved and displayed.

IMPLEMENTATION:

```
#Importing modules
import pandas as pd
import sys
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import random
# This is to supress the warning messages (if any) generated in our code

import os
import re
import nltk
import requests
import warnings
```

```
books = pd.read_csv('Books.csv')
users = pd.read_csv('Users.csv')
ratings = pd.read_csv('Ratings.csv')
```

```
books.head()
```

	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	Image-URL
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images/P/0195153448.
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	http://images.amazon.com/images/P/0060973129.0...	http://images.amazon.com/images/P/0060973129.
3	0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.amazon.com/images/P/0374157065.0...	http://images.amazon.com/images/P/0374157065.
4	0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	http://images.amazon.com/images/P/0393045218.0...	http://images.amazon.com/images/P/0393045218.

```
ratings.head()
```

	User-ID	ISBN	Book-Rating
0	276725	034545104X	0
1	276726	0155061224	5
2	276727	0446520802	0
3	276729	052165615X	3
4	276729	0521795028	6

```
print(books.shape)
print(ratings.shape)
print(users.shape)
```

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

```
books.duplicated().sum()
```

```
0
```

```
ratings.duplicated().sum()
```

```
0
```

```
users.duplicated().sum()
```

```
0
```

Popularity Based Recommender System

```
ratings_with_name = ratings.merge(books,on='ISBN')
```

```
num_rating_df = ratings_with_name.groupby('Book-Title').count()['Book-Rating'].reset_index()
num_rating_df.rename(columns={'Book-Rating':'num_ratings'},inplace=True)
num_rating_df
```

	Book-Title	num_ratings
0	A Light in the Storm: The Civil War Diary of ...	4
1	Always Have Popsicles	1
2	Apple Magic (The Collector's series)	1
3	Ask Lily (Young Women of Faith: Lily Series, ...	1
4	Beyond IBM: Leadership Marketing and Finance ...	1
...	...	...
241066	Ã?Â?lpiraten.	2
241067	Ã?Â?rger mit Produkt X. Roman.	4
241068	Ã?Â?sterlich leben.	1

```
avg_rating_df = ratings_with_name.groupby('Book-Title').mean()['Book-Rating'].reset_index()
avg_rating_df.rename(columns={'Book-Rating':'avg_rating'},inplace=True)
avg_rating_df
```

	Book-Title	avg_rating
0	A Light in the Storm: The Civil War Diary of ...	2.250000
1	Always Have Popsicles	0.000000
2	Apple Magic (The Collector's series)	0.000000
3	Ask Lily (Young Women of Faith: Lily Series, ...	8.000000
4	Beyond IBM: Leadership Marketing and Finance ...	0.000000
...	...	...

```
books['Image-URL-M'][0]
```

```
'http://images.amazon.com/images/P/0195153448.01.MZZZZZZZ.jpg'
```

```
users.head(50)
```

	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
5	6	santa monica, california, usa	61.0
6	7	washington, dc, usa	NaN
7	8	timmins, ontario, canada	NaN
8	9	germantown, tennessee, usa	NaN
9	10	albacete, wisconsin, spain	26.0

```
popular_df = num_rating_df.merge(avg_rating_df,on='Book-Title')
popular_df
```

	Book-Title	num_ratings	avg_rating
0	A Light in the Storm: The Civil War Diary of ...	4	2.250000
1	Always Have Popsicles	1	0.000000
2	Apple Magic (The Collector's series)	1	0.000000
3	Ask Lily (Young Women of Faith: Lily Series, ...	1	8.000000
4	Beyond IBM: Leadership Marketing and Finance ...	1	0.000000
...	...	...	...
241066	Ã?Â?lpiraten.	2	0.000000
241067	Ã?Â?rger mit Produkt X. Roman.	4	5.250000
241068	Ã?Â?sterlich leben.	1	7.000000
241069	Ã?Â?stlich der Berge.	3	2.666667
241070	Ã?Â?thique en toc	2	4.000000

241071 rows × 3 columns

```
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
```

```
users.isnull().sum()
```

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

```
ratings.isnull().sum()
```

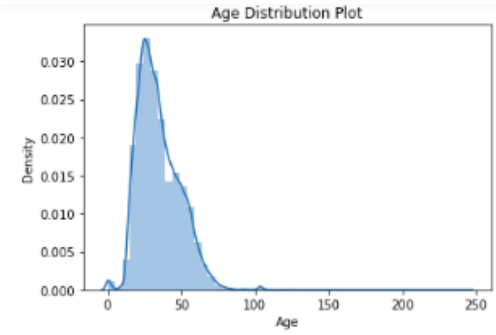
```
User-ID        0
ISBN           0
Book-Rating    0
dtype: int64
```

```
popular_df = popular_df[popular_df['num_ratings']>=250].sort_values('avg_rating',ascending=False).head(50)
```

```
popular_df = popular_df.merge(books,on='Book-Title').drop_duplicates('Book-Title')[['Book-Title','Book-Author','Ima
```

```
popular_df['Image-URL-M'][0]
```

```
'http://images.amazon.com/images/P/0439136350.01.MZZZZZZZ.jpg'
```



Age value's below 5 and above 100 do not make much sense for our book rating case...hence replacing these by NaNs

```
# outlier data became NaN
users.loc[(users.Age > 100) | (users.Age < 5), 'Age'] = np.nan
```

```
users.isna().sum()
```

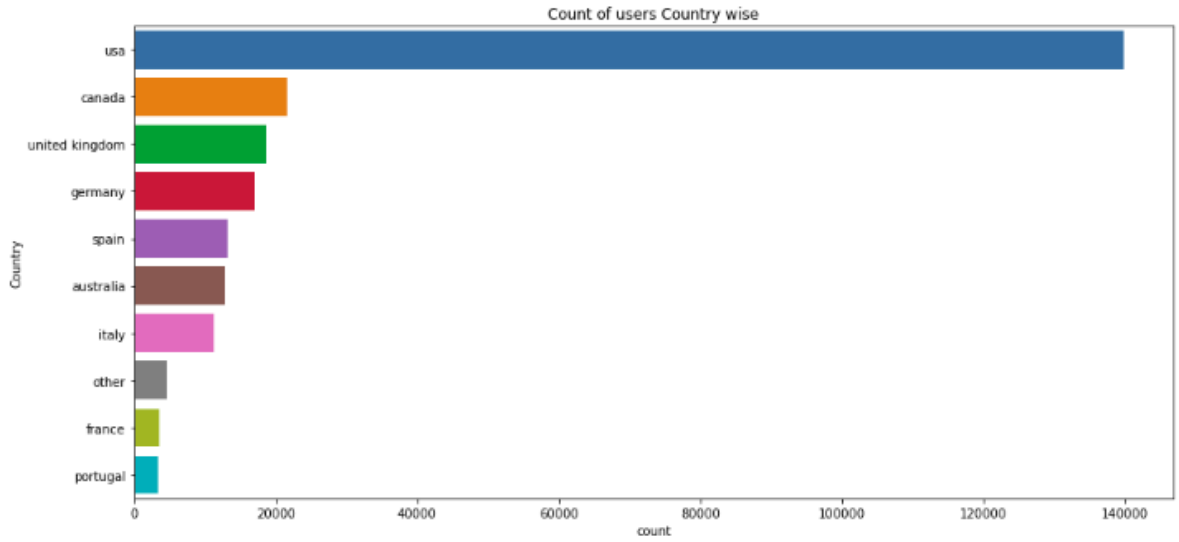
```
User-ID         0
Age        112010
Country         0
dtype: int64
```

Age has positive Skewness (right tail) so we can use median to fill Nan values, but for this we don't like to fill Nan value just for one range of age. To handle this we'll use country column to fill Nan.

```
users['Age'] = users['Age'].fillna(users.groupby('Country')['Age'].transform('median'))
```

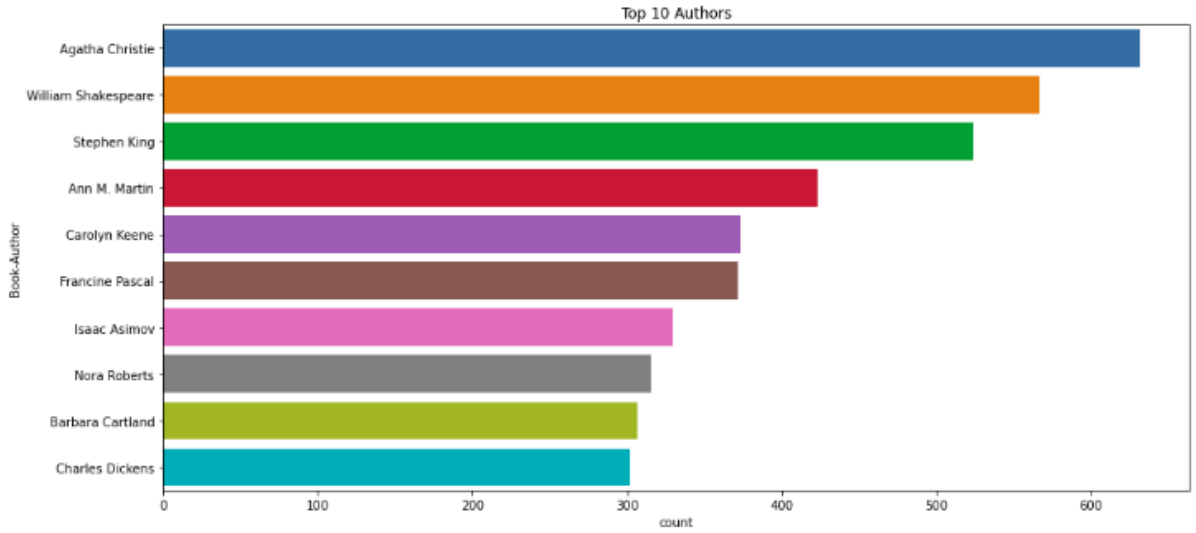
```
plt.figure(figsize=(15,7))
sns.countplot(y='Country',data=users,order=pd.value_counts(users['Country']).iloc[:10].index)
plt.title('Count of users Country wise')
```

```
Text(0.5, 1.0, 'Count of users Country wise')
```



```
plt.figure(figsize=(15,7))
sns.countplot(y='Book-Author',data=books,order=pd.value_counts(books['Book-Author']).iloc[:10].index)
plt.title('Top 10 Authors')
```

```
Text(0.5, 1.0, 'Top 10 Authors')
```




```
plt.figure(figsize=(15,7))
sns.countplot(y='Publisher',data=books,order=pd.value_counts(books['Publisher']).iloc[:10].index)
plt.title('Top 10 Publishers')
```

```
books['Year-Of-Publication']=books['Year-Of-Publication'].astype('str')
a=list(books['Year-Of-Publication'].unique())
a=set(a)
a=list(a)
a = [x for x in a if x is not None]
a.sort()
print(a)
```

```
#investigating the rows having 'DK Publishing Inc' as yearOfPublication
books.loc[books['Year-Of-Publication'] == 'DK Publishing Inc',:]
```

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

```
#From above, it is seen that bookAuthor is incorrectly loaded with bookTitle, hence making required corrections
#ISBN '0789466953'
books.loc[books.ISBN == '0789466953','Year-Of-Publication'] = 2000
books.loc[books.ISBN == '0789466953','Book-Author'] = "James Buckley"
books.loc[books.ISBN == '0789466953','Publisher'] = "DK Publishing Inc"
books.loc[books.ISBN == '0789466953','Book-Title'] = "DK Readers: Creating the X-Men, How Comic Books Come to Life

#ISBN '078946697X'
books.loc[books.ISBN == '078946697X','Year-Of-Publication'] = 2000
books.loc[books.ISBN == '078946697X','Book-Author'] = "Michael Teitelbaum"
books.loc[books.ISBN == '078946697X','Publisher'] = "DK Publishing Inc"
books.loc[books.ISBN == '078946697X','Book-Title'] = "DK Readers: Creating the X-Men, How It All Began (Level 4: Pr

#rechecking
books.loc[(books.ISBN == '0789466953') | (books.ISBN == '078946697X'),:]
#corrections done
```

```
#rechecking
books.loc[(books.ISBN == '0789466953') | (books.ISBN == '078946697X'),:]
#corrections done
```

```
#investigating the rows having 'Gallimard' as yearOfPublication
books.loc[books['Year-Of-Publication'] == 'Gallimard',:]
```

```
#making required corrections as above, keeping other fields intact
books.loc[books.ISBN == '2070426769','Year-Of-Publication'] = 2003
books.loc[books.ISBN == '2070426769','Book-Author'] = "Jean-Marie Gustave Le ClÃ?Â©zio"
books.loc[books.ISBN == '2070426769','Publisher'] = "Gallimard"
books.loc[books.ISBN == '2070426769','Book-Title'] = "Peuple du ciel, suivi de 'Les Bergers"


books.loc[books.ISBN == '2070426769',:]
```

```
books['Year-Of-Publication']=pd.to_numeric(books['Year-Of-Publication'], errors='coerce')

print(sorted(books['Year-Of-Publication'].unique()))
#Now it can be seen that yearOfPublication has all values as integers
```

```
param_grid = {'n_factors': [80,100],
              'n_epochs': [5, 20],
              'lr_all': [0.002, 0.005],
              'reg_all': [0.2, 0.4]}

gs = GridSearchCV(SVD, param_grid, measures=['rmse', 'mae'], cv=3)
gs.fit(data)

print(gs.best_score['rmse'])
print(gs.best_params['rmse'])
```

#Analysis of Collaborative Filtering model results In this part, let's examine in detail the results obtained by the SVD model that provided the best RMSE score.

```
trainset, testset = train_test_split(data, test_size=0.2)

model = SVD(n_factors=80, n_epochs=20, lr_all=0.005, reg_all=0.2)
model.fit(trainset)
predictions = model.test(testset)
```

```
df_pred = pd.DataFrame(predictions, columns=['user_id', 'isbn', 'actual_rating', 'pred_rating', 'details'])
df_pred['impossible'] = df_pred['details'].apply(lambda x: x['was_impossible'])
df_pred['pred_rating_round'] = df_pred['pred_rating'].round()
df_pred['abs_err'] = abs(df_pred['pred_rating'] - df_pred['actual_rating'])
df_pred.drop(['details'], axis=1, inplace=True)
df_pred.sample(5)
```

```
df_ratings_top.rename(columns={'user_id':'userID' ,'isbn':'ISBN','book_rating':'bookRating'},inplace=True)
```

```
df_ratings_top.head()
```

#Implementing KNN

```
#Generating ratings matrix from explicit ratings table
ratings_matrix = df_ratings_top.pivot(index='userID', columns='ISBN', values='bookRating')
userID = ratings_matrix.index
ISBN = ratings_matrix.columns
print(ratings_matrix.shape)
ratings_matrix.head()
#Notice that most of the values are NaN (undefined) implying absence of ratings
```

```
n_users = ratings_matrix.shape[0] #considering only those users who gave explicit ratings
n_books = ratings_matrix.shape[1]
print (n_users, n_books)
```

```
ratings_matrix.fillna(0, inplace = True)
ratings_matrix = ratings_matrix.astype(np.int32)
```

```
#checking first few rows
ratings_matrix.head(5)
```

```
sparsity = 1.0-len(ratings_explicit)/float(ratings_explicit.shape[0]*n_books)
print ('The sparsity level of Book Crossing dataset is ' +  str(sparsity*100) + ' %')
```

```
combine_book_rating = pd.merge(ratings, books, on = 'ISBN')
columns = ['Book-Author','Year-Of-Publication', 'Publisher']
```

```
import sklearn
from sklearn.decomposition import TruncatedSVD

SVD = TruncatedSVD(n_components=12, random_state=17)
matrix = SVD.fit_transform(X)
matrix.shape
```

```
corr = np.corrcoef(matrix)
corr.shape
```

Let's find books similar to Harry Potter and the Sorcerer's Stone (Book 1)

```
us_canada_book_title = us_canada_user_rating_pivot2.columns
us_canada_book_list = list(us_canada_book_title)
coffey_hands = us_canada_book_list.index("1984")
```

```
us_canada_user_rating_matrix = csr_matrix(us_canada_user_rating_pivot.values)
```

Finding the Nearest Neighbors

```
from sklearn.neighbors import NearestNeighbors

model_knn = NearestNeighbors(metric = 'cosine', algorithm = 'brute')
model_knn.fit(us_canada_user_rating_matrix)
```

```
pd.set_option('display.float_format', lambda x: '%.3f' % x)
print(book_ratingcount['TotalRatingCount'].describe())
```

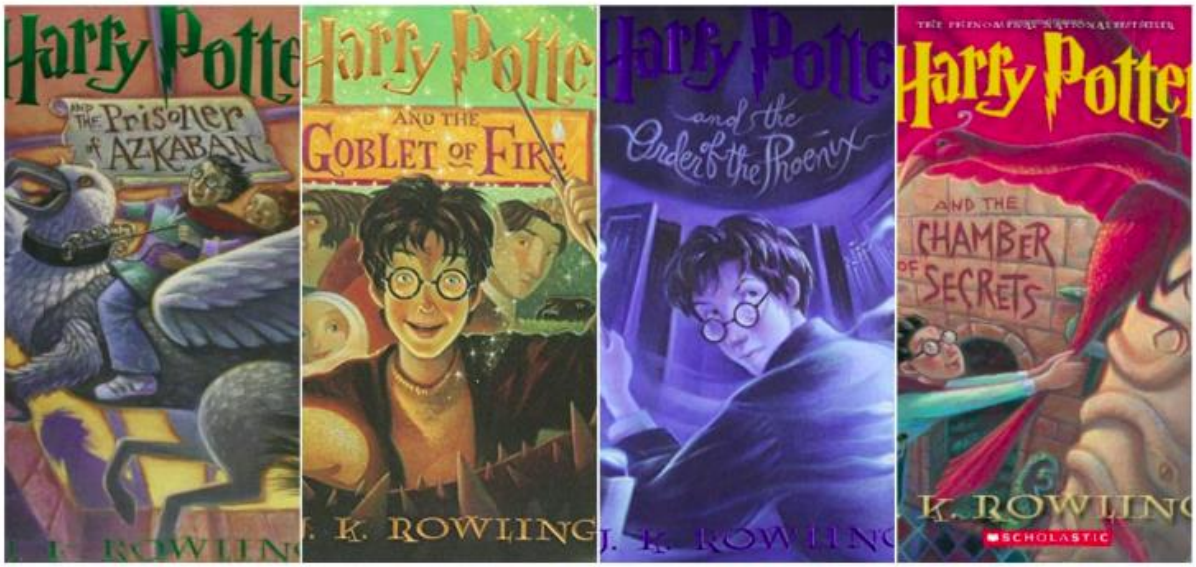
The median book has been rated only once. Let's look at the top of the distribution:

```
print(book_ratingcount['TotalRatingCount'].quantile(np.arange(.9,1,.01)))
```

About 1% of the books received 50 or more ratings. Because we have so many books in our data, we will limit it to the top 1%.

```
corr_coffey_hands  = corr[coffey_hands]
```

```
list(us_canada_book_title[(corr_coffey_hands<1.0) & (corr_coffey_hands>0.9)])
```



Collaborative Filtering Based Recommender System

```
x = ratings_with_name.groupby('User-ID').count()['Book-Rating'] > 200
padhe_likhe_users = x[x].index
```

```
filtered_rating = ratings_with_name[ratings_with_name['User-ID'].isin(padhe_likhe_users)]
```

```
y = filtered_rating.groupby('Book-Title').count()['Book-Rating']>=50
famous_books = y[y].index
```

```
final_ratings = filtered_rating[filtered_rating['Book-Title'].isin(famous_books)]
```

```
pt = final_ratings.pivot_table(index='Book-Title',columns='User-ID',values='Book-Rating')
```

```
pt.fillna(0,inplace=True)
```

```
pt
```

User-ID	254	2276	2766	2977	3363	4017	4385	6251	6323	6543	...	271705	273979	274004	274061	274301	274308	275970	277427	277639	2
Book-Title																					
1984	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	10.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1st to Die: A Novel	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	9.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2nd Chance	0.0	10.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4 Blondes	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
A Bend in the Road	0.0	0.0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
Year of Wonders	0.0	0.0	0.0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	9.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
You Belong To Me	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
Zen and the Art of Motorcycle Maintenance: An Inquiry into Values	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
Zoya	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
\0\" Is for Outlaw"	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	8.0	0.0	0.0	0.0	0.0	

706 rows × 810 columns

```
from sklearn.metrics.pairwise import cosine_similarity
```

```
similarity_scores = cosine_similarity(pt)
```

```
def recommend(book_name):
    # index fetch
    index = np.where(pt.index==book_name)[0][0]
    similar_items = sorted(list(enumerate(similarity_scores[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']]
```

```
pt.index[545]
```

```
"The Handmaid's Tale"
```

```
import pickle
pickle.dump(popular_df,open('popular.pkl','wb'))
```

```
pt.index[545]
```

```
"The Handmaid's Tale"
```

```
import pickle
pickle.dump(popular_df,open('popular.pkl','wb'))
```

```
books.drop_duplicates('Book-Title')
```

	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	http://images.amazon.com/images/P/0060973129.0...	http://images.amazon.com/images
3	0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.amazon.com/images/P/0374157065.0...	http://images.amazon.com/images
4	0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	http://images.amazon.com/images/P/0393045218.0...	http://images.amazon.com/images
...	...	...	...	...	...	...	
271354	0449906736	Flashpoints: Promise and Peril in a New World	Robin Wright	1993	Ballantine Books	http://images.amazon.com/images/P/0449906736.0...	http://images.amazon.com/images
271356	0525447644	From One to One Hundred	Teri Sloat	1991	Dutton Books	http://images.amazon.com/images/P/0525447644.0...	http://images.amazon.com/images
		Lily Dale :					

```
pickle.dump(pt,open('pt.pkl','wb'))
pickle.dump(books,open('books.pkl','wb'))
pickle.dump(similarity_scores,open('similarity_scores.pkl','wb'))
```

```
```

FRONTEND CODE USING FLASK:

PYTHON CODE :

```
from flask import Flask, render_template,request
import pickle
import numpy as np

popular_df = pickle.load(open('popular.pkl','rb'))
pt  = pickle.load(open('pt.pkl','rb'))
books  = pickle.load(open('books.pkl','rb'))
similarity_scores  = pickle.load(open('similarity_scores.pkl','rb'))

app = Flask(__name__)
@app.route('/')
def index():
    return render_template("sample.html",
                           book_name= list(popular_df['Book-Title'].values),
                           author = list(popular_df['Book-Author'].values),
```

```
                    image=list(popular_df['Image-URL-M'].values),
                    votes=list(popular_df['num_ratings'].values),
                    rating=list(popular_df['avg_rate'].values),
                    )

@app.route('/recemmond')
def recemmond_ui():
    return render_template('recemmond.html')

@app.route('/recemmond_books',methods = ['post'])
def recemmond():
    user_input = request.form.get('user_input')
    index = np.where(pt.index == user_input)[0][0]
    similar_items = sorted(list(enumerate(similarity_scores[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)
    print(data)

    return render_template('recemmond.html',data=data)

if __name__ == '__main__':
    app.run(debug=True)
```

sample.html :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
  <!-- Latest compiled and minified CSS -->
<!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"
      integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">

</head>
<style>
:root {
  --gradient: linear-gradient(to left top, #DD2476 10%, #FF512F 90%) !important;
}

body {
  background: #111 !important;
}

.card {
  background: #111;
```

```
  border: 1px solid             #0000CD;
  color: rgba(250, 250, 250, 0.8);
  margin-bottom: 2rem;
box-shadow: rgb(0,0,255) 0px 3px 8px;
}

.btn {
  border: 5px solid             #0000CD;
  border-radius:0px;
  border-image-slice: 1;
  color:#fff;
  background-color:             #0000CD;
  text-decoration: none;
  transition: all .4s ease;
  margin-left : 15px;
  margin-bottom : 2rem;
  font-size:12px;
  font-weight:bold;
}

.btn:hover, .btn:focus {
      font-size:12px;
      background-color :#fff;
    border: 0px solid #fff !important;
  box-shadow: #222 1px 0 10px;
  padding: 10px 18px;



}
.card-title{
  font-size : 18px;
}

.card-subtitle{
font-size : 15px;
}
.card-text{
font-size : 16px;
text-align : center;
}
.bg{
color : #fff;
font-weight: bold;
}

</style>

<body style = "background-color: white;">
    <nav class="navbar" style = "background-color :   #0000CD;">
        <a class = "navbar-brand bg "style="color : white;font-weight: bold;">My Book </a>
        <ul class="nav navbar-nav" style="color:red";>
            <li><a href = "/" class="bg" >HOME</a></li>
            <li><a href="/recemmond" class="bg">RECOMMEND</a></li>
            <li><a class="bg">CONTACT</a></li>
        </ul>
    </nav>
      <div class="container">
          <div class="row">
```

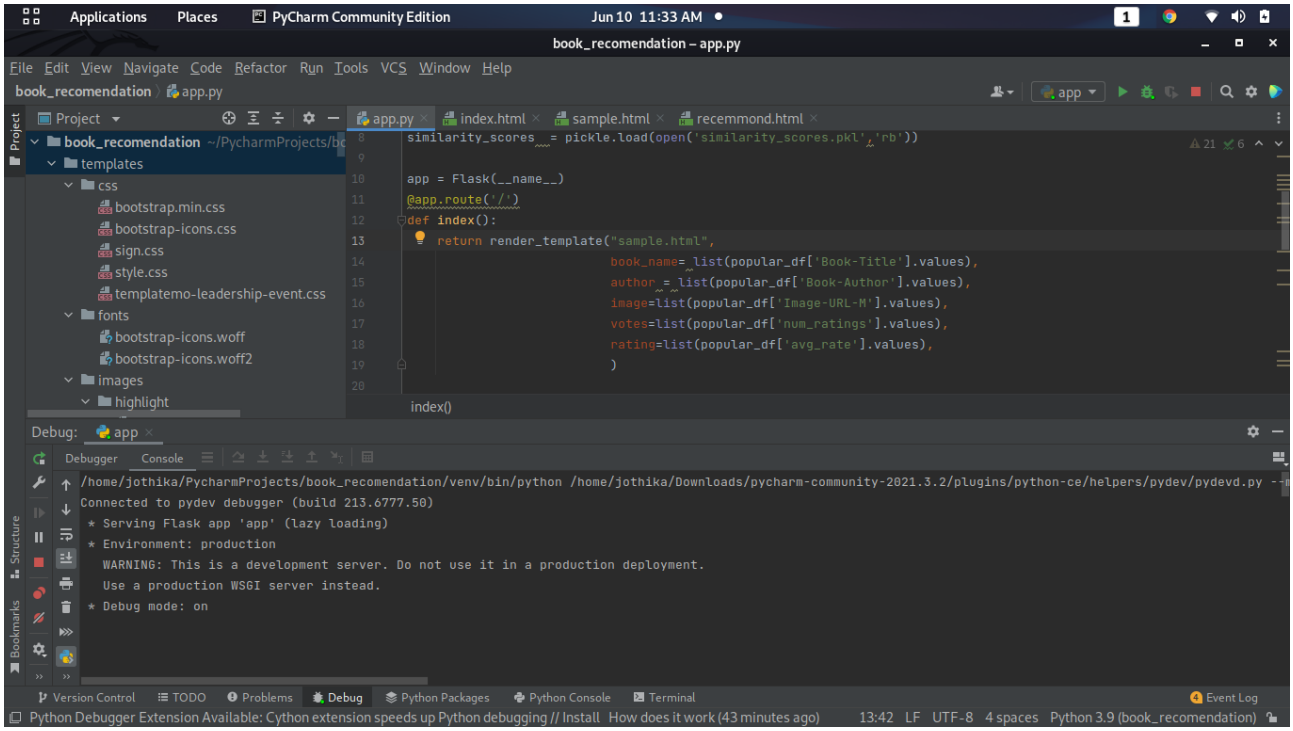
```
            <div class="col-md-12">
               <h1 class="text-white" style="font-size:50px; color:white">Top 50 Books</h1>
            </div>

            {%  for i in range(10)%}
             <div class="container mx-auto mt-4" >
               <div class="row">
                 <div class="col-md-3">
                   <div class="card" >
                  <img src="{{image[i]}}" class="img-responsive center-block d-block " style="margin-top:25px;"
alt="Sample Image">
                 <div class="card-body">
                  <h5 class="card-title " style="margin-left:10px;margin-left:10px;">{{book_name[i]}}</h5>
                     <h6 class="card-subtitle mb-2 text-muted" style="margin-left:10px;">{{author[i]}}</h6>
                   <a href="#" class="btn mr-4"><i class="fas fa-link"></i> VOTES - {{votes[i]}}</a>
                  <a href="#" class="btn "><i class="fab fa-github"></i> RATING - {{rating[i]}}</a>
                 </div>
                 </div>
                  </div>
            {% endfor %}









             </div>
               </div>
          </div>
       </div>




</body>
</html>
```

recommend.html :

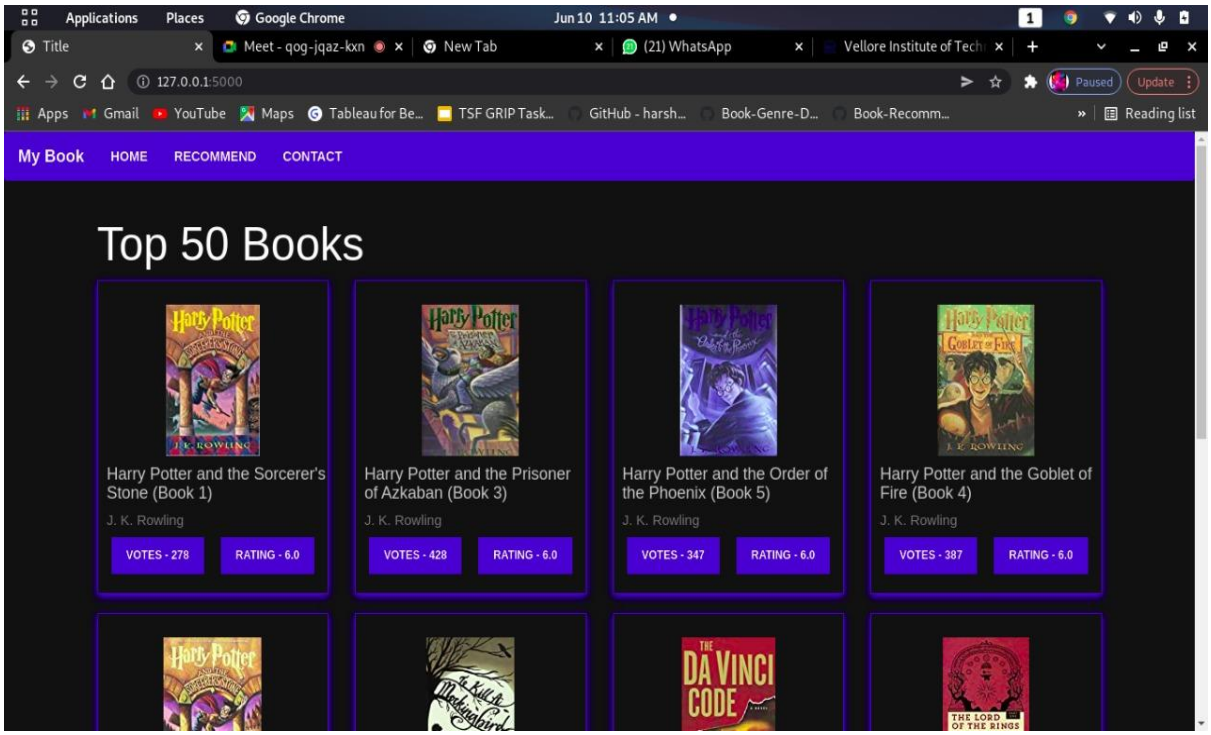
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Book Recmmondation system</title>
    <!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@3.3.7/dist/css/bootstrap.min.css"
      integrity="sha384-
BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
crossorigin="anonymous">
</head>
<style>
:root {
  --gradient: linear-gradient(to left top, #DD2476 10%, #FF512F 90%) !important;
```

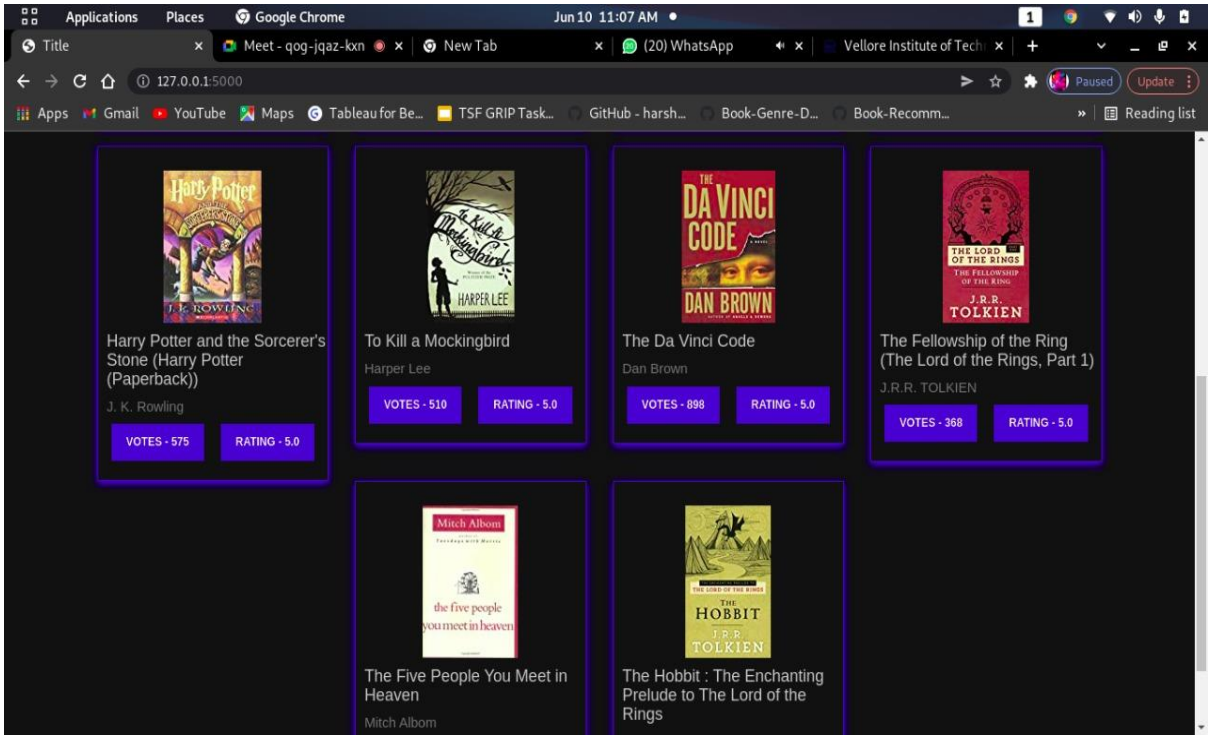
```
}

body {
  background: #111 !important;
}

.card {
  background: #111;
  border: 1px solid             #0000CD;
  color: rgba(250, 250, 250, 0.8);
  margin-bottom: 2rem;
box-shadow: rgb(0,0,255) 0px 3px 8px;
}

.btn {
  border: 5px solid             #0000CD;
  border-radius:0px;
  border-image-slice: 1;
  color:#fff;
  background-color:             #0000CD;
  text-decoration: none;
  transition: all .4s ease;
  margin-left : 15px;
  margin-bottom : 2rem;
  font-size:12px;
  font-weight:bold;
}

.btn:hover, .btn:focus {
      font-size:12px;
      background-color :#fff;
    border: 0px solid #fff !important;
  box-shadow: #222 1px 0 10px;
  padding: 10px 18px;



}
.card-title{
  font-size : 18px;
}

.card-subtitle{
font-size : 15px;
}
.card-text{
font-size : 16px;
text-align : center;
}
.bg{
color : #fff;
font-weight: bold;
}

</style>
<body style = "background-color: black">
    <nav class="navbar" style = "background-color :   #0000CD;">
        <a class = "navbar-brand bg "style="color : white;font-weight: bold;">My Book </a>
        <ul class="nav navbar-nav" style="color:red";>
```

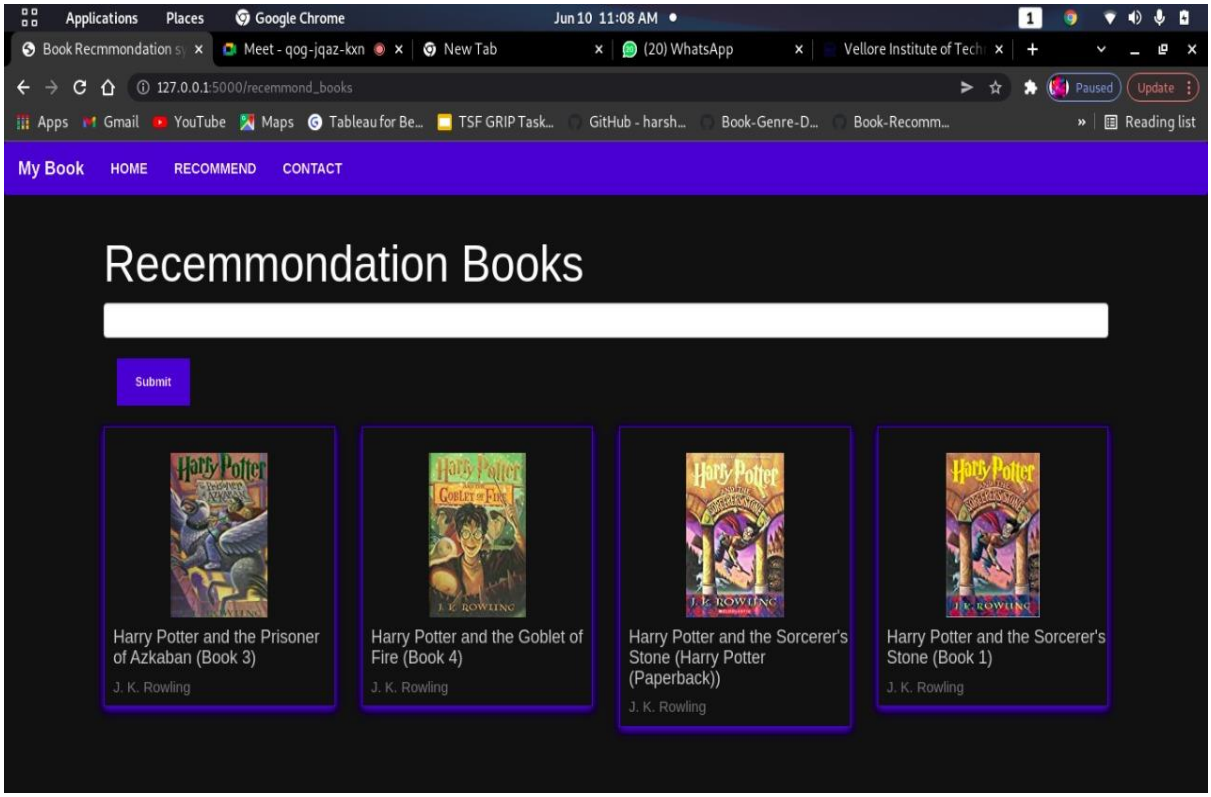
```
            <li><a href = "/" class="bg" >HOME</a></li>
            <li><a href="/recemmond" class="bg">RECOMMEND</a></li>
            <li><a class="bg">CONTACT</a></li>
          </ul>
      </nav>
        <div class="container">
          <div class="row">
            <div class="col-md-12">
              <h1 class="text-white" style="font-size:50px; color :white" >Recemmondation Books</h1>
              <form action = "/recemmond_books" method = "post">
                <input name = "user_input" type = "text" class = "form-control form-control-lg"><br>
                <input type = "submit" class="btn btn-lg btn-warning">
              </form>
            </div>
            {% if data %}
              {% for i in data %}

            <div class="container mx-auto mt-4" >
              <div class="row">
               <div class="col-md-3">
                 <div class="card" >
              <img src="{{i[2]}}" class="img-responsive center-block d-block " style="margin-top:25px;"
alt="Sample Image">
              <div class="card-body">
               <h5 class="card-title " style="margin-left:10px;margin-left:10px;">{{i[0]}}</h5>
                  <h6 class="card-subtitle mb-2 text-muted" style="margin-left:10px;">{{i[1]}}</h6>
              </div>
              </div>
               </div>




                <!--<div class="col-md-3"style="margin-top:50px">
                  <div class="card">
                  <div class="card-body">
                    <img class="card-img-top" src='{{i[2]}}'>
                    <p class="text-white">{{i[0]}}</p>
                    <h4 class="text-white">{{i[1]}}</h4>
                  </div>
                  </div>
                </div>-->
              {% endfor %}
            {% endif %}

        </div>
</body >
</html>
```



RESULT AND DISCUSSION:




Applications
Places
Google Chrome
Jun 10 11:07 AM
Title
Meet - qog-jqaz-kxn
New Tab
(20) WhatsApp
Vellore Institute of Tech
127.0.0.1:5000
Paused
Update
Apps
Gmail
YouTube
Maps
Tableau for Be...
TSF GRIP Task...
GitHub - harsh...
Book-Genre-D...
Book-Recomm...
Reading list
Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))
J. K. Rowling
VOTES - 575
RATING - 5.0
To Kill a Mockingbird
Harper Lee
VOTES - 510
RATING - 5.0
The Da Vinci Code
Dan Brown
VOTES - 898
RATING - 5.0
The Fellowship of the Ring (The Lord of the Rings, Part 1)
J.R.R. TOLKIEN
VOTES - 368
RATING - 5.0
The Five People You Meet in Heaven
Mitch Albom
The Hobbit : The Enchanting Prelude to The Lord of the Rings


Applications
Places
Google Chrome
Jun 10 11:08 AM
Book Recmmondation sy
Meet - qog-jqaz-kxn
New Tab
(20) WhatsApp
Vellore Institute of Tech
127.0.0.1:5000/recommond_books
Paused
Update
Apps
Gmail
YouTube
Maps
Tableau for Be...
TSF GRIP Task...
GitHub - harsh...
Book-Genre-D...
Book-Recomm...
Reading list
My Book
HOME
RECOMMEND
CONTACT
Recmmondation Books
Submit
Harry Potter and the Prisoner of Azkaban (Book 3)
J. K. Rowling
Harry Potter and the Goblet of Fire (Book 4)
J. K. Rowling
Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))
J. K. Rowling
Harry Potter and the Sorcerer's Stone (Book 1)
J. K. Rowling

CONCLUSION:

To conclude about our project, we have made analysis of different research papers and algorithm implemented in it about recommendation systems. In our project we have improvised and modified the recommendation systems. This Book Recommendation System has considered many parameters like ratings, book name, book cover images, author etc,

We successfully implemented and found the similar books from Amazon and Flip kart using cosine similarity. Also, the recommendation system was implemented using book covers dataset and Collaborative Filtering algorithm to display most similar book covers based on the input book cover.

Advantages of the Project:

- This Book Recommendation System upon its implementation, No Need for Feature Engineering: Feature engineering is the process of extracting features from raw data to better describe the underlying problem.

- It is a fundamental job in machine learning as it improves model accuracy. The process can sometimes require domain knowledge about a given problem.

FUTURE WORK:

✧ In the future work, there are many different methods which are used in mining the data and can be used for recommendation process. One of the algorithm which can be used in the future work is K-Means clustering, Sparks ALS (Alternating Least Squares) Algorithm.

- ✧ In our future work, we shall propose a suggestion system for recommending online courses , system for recommending movies using the convolutional neural network (CNN).
- ✧ The proposed work can be used to suggest items such as music, and other products in other domains.

REFERENCE:

[1] Tewari, A. S., & Priyanka, K. (2018, November). Book recommendation system based on collaborative filtering and association rule mining for college students. In *2018 International Conference on Contemporary Computing and Informatics (IC3I)* (pp. 135- 138). IEEE.

[2] Anandaraj, A., Ram, P. Y., Kumar, K. S. R., Revanth, M., & Praveen, R. (2021, March). Book Recommendation System with TensorFlow. In *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)* (Vol. 1, pp. 1665-1669). IEEE.

[3]Anwar, T., & Uma, V. (2020, October). Book Recommendation for eLearning Using Collaborative Filtering and Sequential Pattern Mining. In *2020 International Conference on Data Analytics for Business and Industry:* IEEE.

[4] Rana, A., & Deeba, K. (2019, November). Online book recommendation system using collaborative filtering (with Jaccard similarity). In *Journal of Physics: Conference Series* (Vol. 1362, No. 1, p. 012130). IOP Publishing.

[5] Tsuji, K., Yoshikane, F., Sato, S., & Itsumura, H. (2020, August). Book recommendation using machine learning methods based on library loan records and bibliographic information. In *2020 IIAI 3rd International Conference on Advanced Applied Informatics* (pp.76-79). IEEE.

[6] Rajpurkar, S., Bhatt, D., Malhotra, P., Rajpurkar, M. S. S., & Bhatt, M. D. R. (2018). Book recommendation system. *IJIRST–International Journal for Innovative Research in Science & Technology*, *1*(11), 314-316.

[7] Cui, B., & Chen, X. (2020, August). An online book recommendation system based on web service. In *2020 Sixth International Conference on Fuzzy Systems and Knowledge Discovery* (Vol. 7, pp. 520-524). IEEE

[8] Puritat, K., & Intawong, K. (2020, March). Development of an open source automated library system with book recommedation system for small libraries. In *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)* (pp. 128-132). IEEE.

[9] Tewari, A. S., Kumar, A., & Barman, A. G. (2019, February). Book recommendation system based on combine features of content based filtering, collaborative filtering and association rule mining. In *2019 IEEE International Advance Computing Conference (IACC)* (pp. 500-503). IEEE.

[10] Sahu, S. P., Nautiyal, A., & Prasad, M. (2017). Machine learning algorithms for recommender system—a comparative analysis. *International Journal of Computer Applications Technology and Research*, *6*(2), 97-100.