Intelligent Bookshelf Analysis System

A Project Report Submitted
In Partial Fulfillment of the Requirements for
Degree of Bachelor of Technology

in

COMPUTER SCIENCE AND ENGINEERING

by

Rekhansh Gupta(2101330100190)
Abhinav Gupta (2101330100007)
Arun Chauhan (2101330100071)

Under the Supervision of
Mr. Rahul Kumar Sharma
Associate Professor, CSE, NIET

Department of Computer Science and Engineering
School of Computer Science & Information Technology
NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY, GREATER NOIDA
(An Autonomous Institute)
Affiliated to DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW
June, 2025

DECLARATION

We hereby declare that the work presented in this report entitled "**Intelligent Bookshelf Analysis System**", was carried out by us at Department of Computer Science & Engineering, Noida Institute of Engineering and Technology (an Autonomous Institute) affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow. We have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute. We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. We have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

We affirm that no portion of our work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, we shall be fully responsible and answerable.

Name : Abhinav Gupta

Roll Number : 2101330100007

(Candidate Signature)

Name : Rekhansh Gupta

Roll Number : 2101330100190

(Candidate Signature)

Name : Arun Chauhan

Roll Number : 2101330100071

(Candidate Signature)

CERTIFICATE

Certified that **Rekhansh Gupta (2101330100190)**, **Abhinav Gupta (2101330100009)**, **Arun Chauhan (2101330100071)** have carried out the research work presented in this Project Report entitled "**Intelligent Bookshelf Analysis System"** for the award of **Bachelor of Technology**, **Computer Science & Engineering** from Dr. APJ Abdul Kalam Technical University, Lucknow under our supervision. The Project Report embodies results of original work, and studies are carried out by the students himself. The contents of the Project Report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature	Signature
Mr. Rahul Kumar Sharma	Dr. Kumud Saxsena
Assistant Professor	HoD
CSE	CSE
NIET Greater Noida	NIET Greater Noida

Date :

ACKNOWLEDGEMENTS

We would like to express my gratitude towards Mr. Rahul Kumar Sharma for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

Our thanks and appreciations to respected HOD, Dy. HOD, for their motivation and support throughout.

We would like to express our gratitude and indebtedness to my supervisors **Mr. Rahul Kumar Sharma, Assistant Professor, CSE, NIET** for their valuable guidance, encouragement, continuous inspiration, and valuable suggestions throughout our project work. We are heavily indebted to him for imparting his priceless and meticulous supervision at each phase of project work whenever needed. He inspired us in innumerable ways to accomplishing this project work. Thanks is due to him not only for the academic guidance but also for the moral support, consistent kindness and never-ending encouragement given by him/her.

We also express our thanks to all the faculty and staff members of Department of Computer Science & Engineering for their continuous encouragement. Our thanks and appreciations to respected HOD, CSE, Dr. Kumud Saxena for her motivation and support throughout.

We are privileged to fulfil our parents' ambition and greatly indebted to them for providing us moral support and good wishes in the completion of this project work. Finally, we wish to express our gratitude to all those who have directly or indirectly assisted us & whose help and cooperation made this research work accomplished.

ABSTRACT

Today the World Wide Web provides users with a vast array of information, and commercial activity on the Web has increased to the point where hundreds of new companies are adding web pages daily. This has led to the problem of information overload. Recommender systems have been developed to overcome this problem by providing recommendations that help individual users identify content of interest by using the opinions of a community of users and/or the user's preferences.

The aim of this thesis was to design and evaluate different approaches for producing personalised recommendations within the book domain. To achieve this goal, the project first investigated existing recommender systems and profiling techniques.
The next step was to build users' profiles by monitoring users' behaviour, and develop three different approaches for producing recommendations. Finally, an evaluation of the system recommendations' accuracy was done, by first conducting live user experiments and then performing offline analysis to measure the recommendations' accuracy using appropriate methods for testing.

The system evaluation results show that the accuracy of the system recommendations is very good and that a recommender system based on the combination of content-based and collaborative filtering approaches provides more accurate recommendations for the book domain.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
DL	Deep learning
LDA	Latent Dirichlet allocation
LSTM	Long short-term memory
GRU	Gated Recurrent Unit
NLP	Natural language processing
TF-IDF	Term Frequency-Inverse Document Frequency
GloVe	Global Vectors
CURB	Scalable Online Algorithm
EANN	Event Adversarial Neural Network
BiLSTM	Bidirectional LSTM
CNN	Convolutional neural network
MLP	Multilayer perceptron
API	Application programming interface
NB	Naive Bayes
CNN	Convolution neural network
NER	Named Entity Recognition
KNN	K-Nearest Neighbours

CHAPTER 1
INTRODUCTION

This is an experimental project which first, designs…. And second evaluates different approaches for offering recommendations to readers regarding books they may wish to purchase, as part of an online bookshop website.

Today the World Wide Web has provided access to a vast array of information through the web pages, as a result of the Internet growth. Also, commercial activity on the Web has increased to the point where hundreds of new companies are adding Web pages daily. With this increase in the information sources, a problem of information overload occurs, in which the users are trying to deal with an excess of information that is not useful to them as they try to make sensible decisions (Losee, 1989). As a response to this problem, a range of tools to help with retrieving, searching, and filtering have been developed.

The tool most widely used to alleviate the problem of information overload is the search engine. The benefits for the users from search engine technology have decreased as the number of web pages has grown. In addition, the user must first consider the large number of search tools available and decide which one to access. Then the user must interact with each one individually because search engines are typically not personalised to individual users or their prevailing context. Users usually make a choice on the basis of their personal experience or other people's experience. Based on these facts, recommender systems have been developed to provide recommendations that help individual users identify content of interest by using the opinions of a community of users and/or the user's preferences.

Today various recommendation systems play an important role in supporting commercial websites to help users find items that they know they would like to purchase, as well as discover new items about which they had been unaware. The ability to persuade the consumers to buy a suitable item is a significant goal for any recommender system in an ecommerce environment. However, for any recommender system to be successful, the consumer must trust and accept the system's recommendations. This is done with a clear explanation from the system, presented in a way that is in keeping with the consumer's preferences. A good recommender system can significantly contribute to achieving the consumer's acceptance of the system recommendations.

1.1 PROBLEM DEFINITION

This project aims to design and evaluate different approaches for computing recommendations within the book domain to provide personalised recommendations to the users.

1.2 SCOPE AND OBJECTIVES

- An effective solution to the issue of information overload in e-commerce websites is the recommender system.
- This method offers users accurate recommendations.
- most reliable book-related suggestion technology.

Objectives:

- Look into and assess the profiling and recommender systems that are already in use.
- By observing dynamic user behaviours, you can create a user's profile for a recommender system. The user profile needs to change to reflect the user's shifting interests.
- Create a recommender system that uses a variety of computation methods.
- Utilize the right methods to assess the system's recommendations' accuracy.

CHAPTER 2
LITERATURE SURVEY

An idea for Content-Based Book Recommending Using ML Tools for Text Categorization was put forth by Raymond J. Mooney and Loriene Roy. They outline a content-based book recommendation system that classifies text using machine learning and information extraction. Item recommendations are made based on information about the item itself rather than on the preferences of other users. On the other hand, learning customiZed profiles from descriptions of examples enables a system to uniquely characteriZe each customer without the need to match his or her interests to another's. Through the use of automatic text-categorization techniques on semi-structured text downloaded from the web, they have been investigating content-based book recommendations. A database of book data is used by the present prototype system, called LIBRA (Learning Intelligent Book Recommending Agent). After employing a Bayesian learning algorithm to learn about the user's profile, the system generates a ranked list of the most highly suggested additional books from its library. Even when the method provides very modest training sets, the overall results are quite positive. An original content-based book recommender called LIBRA makes use of a straightforward.

A unique book recommendation system was proposed by Binge Cui and Xin Chen. When readers are unable to locate the desired book using the library's bibliographic retrieval system, they are directed to the recommendation pages. It is a web-based system for recommending books to a library's patrons. After logging in, a user can search for books using author names or keywords like book titles. A bibliographic retrieval system will then look for books using the same keywords. If the recommendation system returns any results, submit these keywords to the web books retrieval module. Web Books Retrieval Module allows the librarian or administrator of the online book recommendation system to search the online bookshop using keywords by creating accounts on sites like Amazon. As a result, the web retrieval module searches these online bookshops as the logged-in user when the keyword is presented to it. The user will receive the results from these online booksellers in the form of recommendations. The statistic and analysis module will determine the value of that specific book based on user recommendations. The Auto-Order Module will then generate a book order automatically based on the analysis results according to this value of book. The Short Message and Email Notification Module will get a report from the Book Storage System once the purchased books have been shelved (fig 2.1). Then, utilizing a message and email server, it will inform the readers who have suggested the books that have been acquired.

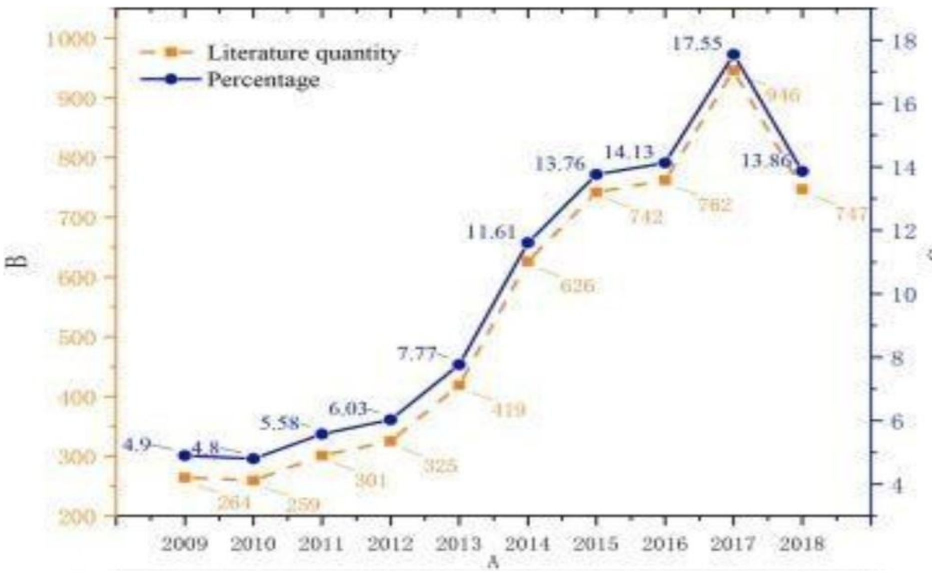


Fig 2.1 Basic Statistics

An effective and best-in-class hybrid recommendation algorithm was proposed by Dharmendra Pathak, Sandeep Matharia, and C. N. S. Murthy , and it provides recommendations that are more in line with user preferences. The hybrid recommendation in this case combines collaborative, content-based, and contextbased algorithms. The primary input for collaborative filtering is rating, or the votes of numerous users, content-based data, which is user-specific information like interests, dates of birth, and priorities, and context-based data, which is behavioural information like time, taste, mood, and weather. The similarity is measured using the cosine similarity. According to the user's prior history, there are subject priority. When they buy a book, do they check to see if the subject priority is changed from what was previously set? If so, topic priority 3 and subject priority 2 should be reset. Priority 1 will remain the same. They came to the conclusion that the proposed Hybrid book recommendation algorithm is superior to the others based on calculations and results.

A book recommendation system based on integrated features of content filtering, collaborative filtering, and association rule mining was proposed by Anand Shanker Tewari, Abhay Kumar, and Asim Gopal Barman. When a customer searches for a book, the search is recorded in the customer's purchase or search history. Recommendation services conduct some filtering when a customer is not online, and the results are saved in the customer's web profile. The recommendations will be created automatically the following time the customer visits the website. Web Usage Mining (WUM) is used in content-based filtering to give customers the pertinent information they need. web server access logs, browser caches, or proxy logs are some examples of historical data that web Usage Mining (WUM) commonly uses to extract knowledge. web use Internet user behavior is recorded by mining, which then processes the information (fig 2.2). The item-based collaborative recommendation algorithm and cosine similarity are both used to measure similarity. Compare the outcomes of collaborative content filtering with association rule mining.

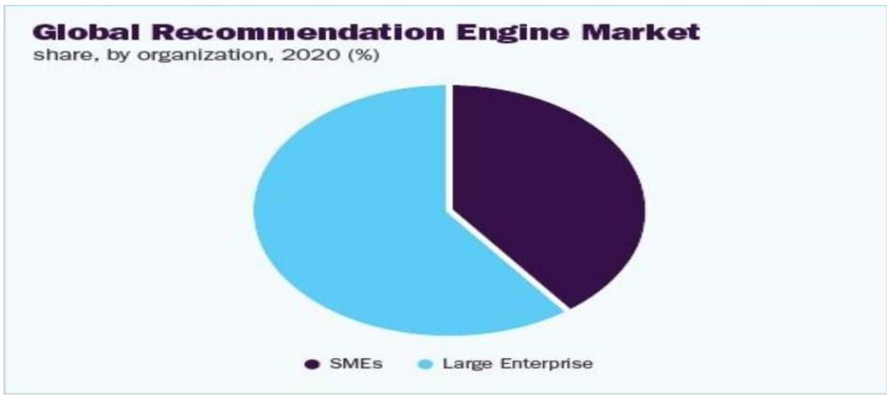


Fig 2.2 Global recommendation system

2.1 INFERENCES FROM LITREATURE SURVEY

For a book recommendation system, Adli Ihsan Hariadi and Dade Nurjanah presented a hybrid-based approach that blends attribute-based and user personality-based methodologies. The MSV-MSL (Most Similar Visited Material to the Most Similar Learner) method is used in this study, and the authors claim that it is the best hybrid attributes-based strategy. When forming neighbourhood ties, the personality trait is used to compare users. The hybrid attribute will use the similarity scores between a target book and its neighbours as well as between the active user and that user's neighbours to generate the recommendation scores of rated books from neighbours. the score for user u's book B, designated as score b. The goal of this is to match up the most similar learner with the most similar visited material. It makes advantage of the collaborative and content values. Utilize the hybrid result as a recommendation after that. That is the webpage that the most similar learner has visited, according to data.

2.2 MOTIVATION

Due to the expansion of the Internet, the World Wide Web now offers access to a wide variety of information via web sites. Additionally, business activity online has grown to the point that every day, hundreds of new businesses add new Web pages. Due to the increase in information sources, a problem known as information overload arises, where users must cope with an abundance of information that is unhelpful to them in order to make sound judgments (Losee, 1989). A variety of tools for accessing, finding, and filtering information have been created as a solution to this issue The search engine is the resource that is most frequently utilised to address the issue of information overload. As the amount of web pages has increased, so too have the benefits for consumers of search engine technology. Additionally, the user must choose which search tool to utilise after carefully weighing the several options. Then, as search engines are often not tailored to specific individuals or their current environment, the user must interact with each one separately. Users typically base their decisions on either their own or other people's experiences. These facts led to the development of recommender systems, which leverage user feedback from a community of users and/or the user's own preferences to generate recommendations that assist individual users in identifying content of interest.

2.3 OPEN PROBLEMS IN EXISTING SYSTEM

The content of the Web is expanding by an estimated 170,000 pages every day, with an Internet growth rate of at least 10% per month (Turban et al., 2000). Due to the abundance of information available to consumers on the World Wide
Web, it can be challenging for them to pick exactly what they want. Information overload is a condition when consumers strive to manage more information but are unable to make rational decisions (Losee, 1989).

A variety of retrieval, searching, and filtering techniques have been created to aid with the problem of information overload. The search engine is the most frequently used instrument to help with the issue of information overload.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES / RISK ANALYSIS OF THE PROJECT

The project will create and assess a collaborative filtering and content-based recommender system for a real online bookstore. Machine learning methods are typically needed for content-based recommendations in order to identify trends in the products customers like (Middleton, 2003). The experiences of actual users will be reflected in the content-based technology. Users' profiles will be created so that their behaviour may be tracked. Additionally, the system will produce recommendations by comparing the contents of the books in the user's profile with those that the user hasn't reviewed.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

3.2.1 Hardware Requirements

• System	**:** Intel® Core™ i5-9300H CPU @ 2.40GHz.
• Monitor	**:** LED.
• Mouse	**:** Logitech.
• Ram	**:** 8.00 GB or above 8.00 GB
• Hard Disk	**:** 1 TB

3.2.2 Software Requirements:

• Operating System	**:** Windows 10, Kali Linux
• Language	**:** Python 3 □
• Framework	**:** Flask 2.2.2

3.2.3 Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

3.2.4 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small-Talk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

3.2.5 Python Features

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

3.2.6 Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

3.2.7 Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

3.2.8 First Python Program

Let us execute programs in different modes of programming.

Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

```
$ python

Python 2.4.3(#1, Nov112010,13:34:43)

[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2

Type "help", "copyright", "credits" or "license" for more information.

>>>
```

Type the following text at the Python prompt and press the Enter –

```
>>>print "Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!").** However, in Python version 2.4.3, this produces the following result –

```
Hello, Python!
```

3.2.9 Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**.

Type the following source code in a test.py file –

```
Print "Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

The application will be developed using the incremental development methodology and will be made up of four increments: Front End, Learning module, Recommendation module and Database increment. The requirements outlined in the Requirements Document will be mapped to manageable increments.

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

4.1.1 Methodology

Recommender systems have been developed to overcome the above mentioned limitations of searching through the massive volume of information available. Recommender systems, in comparison with other filtering tools, require less experience on the part of the user and less effort to specify their interests when querying and operating the system (Resnick and Varian, 1997).
Recommendations systems rely on different technologies for computing recommendations. The most important approaches are content-based filtering and collaborative filtering. Content-based filtering displays users as individuals, while recommender systems employing the collaborative

filtering approach display the user as a part of a group (Fasli, 2006). In addition, an advanced recommender system that combines content-based and collaborative filtering to avoid the limitations of each approach, is called a hybrid approach.

4.1.2MODULE DESCRIPTION

Content based filtering approach

The content-based filtering approach identifies the similarity between a user and the new items using the content of the previously evaluated items in the user profile. In addition, each item in a user profile is characterized by a set of attributes which is constructed by extracting a set of features from an item. Such a profile is used to determine if the new item is similar to the item that a user has preferred in the past. For instance, the Newsweeder is a netnewsfiltering system that suggests news articles to the user based on the user's profile (Lang, 1995). Most content-based approaches are performed on textual documents, such as web pages and articles. The textual document can be easily broken down into individual words, unlike video and physical resources, which required sophisticated analysis.

Collaborative filtering approach

Collaborative filtering recommendations are based on the opinions of a community of similar users. The basic idea is that users recommend items to one another. Collaborative filtering makes this possible by asking the users to rate items, which allows the system to recommend new items that similar users have rated highly. For instance, MovieLens is a movie recommender system that uses collaborative filtering to help people find movies they will like in the huge stream of available movies. Collaborative filtering works well for multimedia technology such as music and movies.

Data Set

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).

Recommender systems are really critical in some industries as they can generate a huge amount of income when they are efficient or also be a way to stand out significantly from competitors. As a

proof of the importance of recommender systems, we can mention that, a few years ago, Netflix organised a challenges (the
“Netflix prize”) where the goal was to produce a recommender system that performs better than its own algorithm with a prize of 1 million dollars to win

4.2 DATA SET

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 case of several authors, only the first is provided. URLs linking to cover images are also given, appearing in three different flavours (ImageURL-S, Image-URL-M, Image-URL-L), i.e., small, medium, large. These URLs point to the Amazon web site.

4.2.1 Recommender System

Recommender systems intend to provide users with suggestions of items that they may be interested in, based upon their past preferences, history of purchase, or demographic information, as well as the environment of possible items. In addition, a recommender system helps the site adapt itself and provide individual personalisation for each consumer; this increases the sales for the commercial site.

Different forms for providing recommendations have been developed; they can be classified into the following forms: attribute-based recommendations, item-to-item correlation, peopleto- people correlation and non-personalised recommendations (Konstan et al., 2001). For more detailed descriptions.

Recommendations systems rely on different technologies for computing recommendations. The most important approaches are content-based filtering and collaborative filtering. Content-based filtering displays users as individuals, while recommender systems employing the collaborative filtering approach display the user as a part of a group (Fasli, 2006). In addition, an advanced recommender system that combines content-based and collaborative filtering to avoid the limitations of each approach, is called a hybrid approach.

4.2.2 Content based filtering approach

The content-based filtering approach identifies the similarity between a user and the new items using the content of the previously evaluated items in the user profile. In addition, each item in a user profile is characterized by a set of attributes which is constructed by extracting a set of features from an item. Such a profile is used to determine if the new item is similar to the item that a user has preferred in the past. For instance, the Newsweeder is a netnewsfiltering system that suggests news articles to the user based on the user’s profile (Lang, 1995). Most content-based approaches are performed on textual documents, such as web pages and articles. The textual document can be easily broken down into individual words, unlike video and physical resources, which required sophisticated analysis.

Content-based filtering has some shortcomings in recommending items. A user's selection is based on the subjective attributes (such as the quality) of the item (Goldberg et al., 1992); in contrast, content based approaches are based on objective attributes (such as the description of an item) about the items. Also, some items the users may be interested in cannot be recommended to them because content-based methods compare new items with the items previously seen by the user, while the user's interests may be beyond the scope of the previously seen items. Finally, multimedia technology such as sound, video or physical items cannot be analysed automatically for relevant attribute information, due to limitations of resources (Jennings et al., 2005).

4.2.3 Collaborative Filtering approach

Collaborative filtering recommendations are based on the opinions of a community of similar users. The basic idea is that users recommend items to one another. Collaborative filtering makes this possible by asking the users to rate items, which allows the system to recommend new items that similar users have rated highly. For instance, MovieLens is a movie recommender system that uses collaborative filtering to help people find movies they will like in the huge stream of available movies. Collaborative filtering works well for multimedia technology such as music and movies. However, it also has some limitations:

New user problem: A new user starts off with a profile of interests from scratch. The system needs to know the user preferences in different items to generate accurate recommendations.

Cold start problem: New items cannot be recommended until more information is obtained when another user either rates an item or provides feedback on the item (Fasli, 2006). As a result, the recommendations generated by the system will not recommend items similar enough to the users’ interests.

Scalability: A collaborative filtering algorithm should address the scalability issue as the number of users increase and their collective profile size becomes large (Fasli, 2006).

The schematic diagram of the collaborative filtering process is showed in **Figure 4.1**. As you can see from the figure, there is a list of users denoted by U= {u1, u2,…,um} and a list of items I={i1,i2,….,in}. Each user has a list of items. The collaborative filtering algorithm will generate recommendations(fig 4.1), a list of N items that the active user will mostly like, according to the active user. Also, the process will output a prediction, which is the result prediction on item j for the active user (Sarwar et al., 2001).

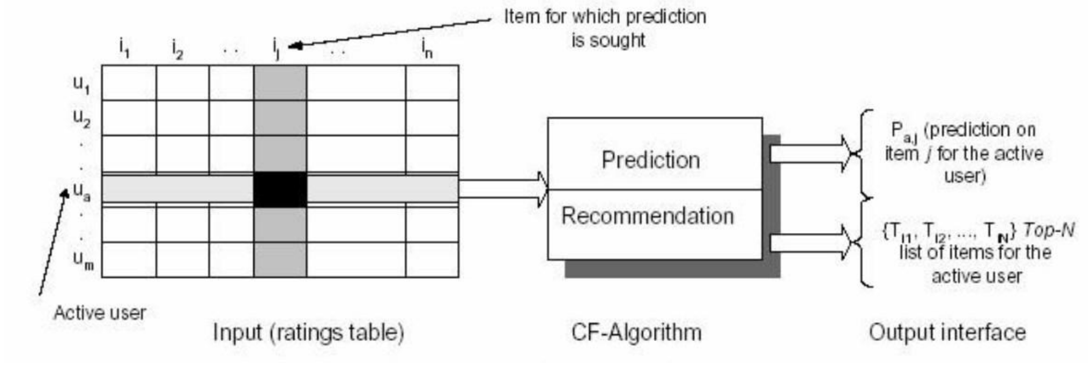

Item for which prediction is sought
i_1 i_2 . . i_j . . i_n
u_1 u_2 . . u_a . . u_m
Active user
Input (ratings table)
Prediction
Recommendation
CF-Algorithm
$P_{a,j}$ (prediction on item j for the active user)
$\{T_{i1}, T_{i2}, ..., T_{iN}\}$ Top-N list of items for the active user
Output interface

Fig 4.1 Collaborative Filtering Process

4.2.4 Hybrid Approach

Hybrid approach introduced to combine the advantages of both content-based and collaborative filtering techniques help to overcome their limitations. These use thestrength of one set of filtering techniques to overcome the limitation of the other.

The hybrid filtering approach is also called “collaborative via content” because content-based profiles are also taken when identifying the similarities among users for collaborative recommendations (Pazzani, 1999).

4.2.5 User - Based collaborative Filtering

User-based algorithm is based on the fact that each user belongs to a larger group of similarly behaving individuals. It uses statistical techniques to find a set of users with similar interests, known as neighbours, in the entire user-item database, to generate a list of recommendation for the active user (Middleton, 2003).

Different measures of similarity that are based on neighbourhood algorithms are used to compute the similarity between the active user and other users in the database, such as the Pearson correlation coefficient and Mean squared differences algorithms (Breese et al., 1998). Moreover, to predict the rating of an item given by the active user, the ratings from the most similar users for the item are averaged and weighted by their similarities to the active user. The Pearson Correlation (fig 4.2) reflects the degree of linear relationship between two variables and ranges from +1 to -1. A positive correlation means that the two users have very similar tastes, while a negative correlation indicates that the users have dissimilar tastes (Fasli, 2006). The Pearson Correlation Coefficient method defines the similarity between two users by:

$$r_{xy} = \frac{\sum_{i=1}^{N} (U_{xi} - \bar{U}_x) * (U_{yi} - \bar{U}_y)}{\sqrt{\sum_{i=1}^{N} (U_{xi} - \bar{U}_x)^2 * \sum_{i=1}^{N} (U_{yi} - \bar{U}_y)^2}}$$

r_{xy}	pearson-r correlation between user x and y
N	number of ratings
U_{xi}	i^{th} rating for user x
$\bar{U}_x$	mean rating for user x

Fig 4.2: Pearson correlation

4.2.6 Item Based collaborative Filtering

The item-based algorithms are developed to overcome the scalability on user-based recommendations. Unlike a user-based approach, the item-based approach identifies the set of items that are similar or related to the item that the active user has evaluated. After that, it computes the similarity between items and then selects the most similar items to the target item within the set of items that the user has rated (Sarwar et al., 2001).

ALGORITHM

- Content based filtering mechanism
- Collaborative based filtering algorithm
- Cosine similarity

4.3 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

Fig 4.3 depicts the architecture of the System

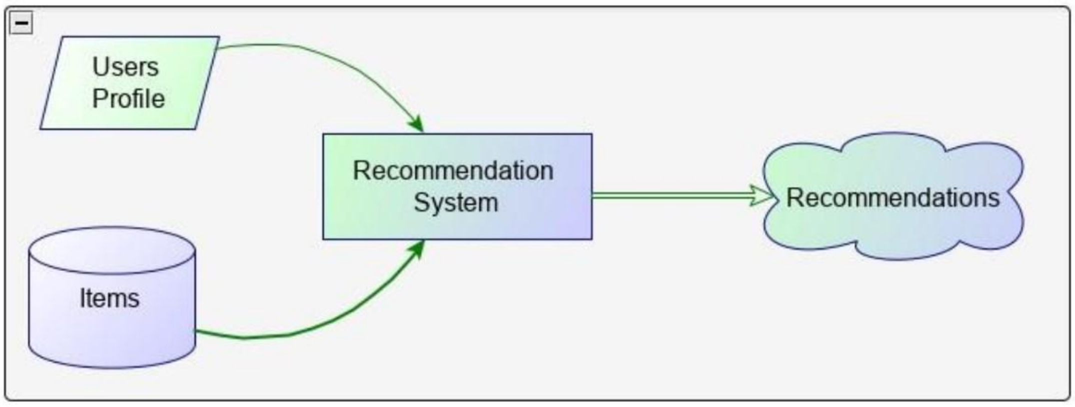


Fig 4.3: System Architecture

4.4 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table 4.1 summarizes different http methods –

Table 4.1: Methods and description of the Flask

S.No	Methods & Description
1	**GET** Sends data in unencrypted form to the server. Most common method.

2	**HEAD** Same as GET, but without response body
3	**POST** Used to send HTML form data to server. Data received by POST method is not cached by server.
4	**PUT** Replaces all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route ()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

```
<html>

<body>

<form action=”http://localhost:5000/login”
method="post">

<p>Enter Name:</p>

<p><input type="text" name="nm"/></p>

<p><input type="submit" value="submit"/></p>
```

```
</form>

</body>

</html>
```

Now enter the following script in Python shell.

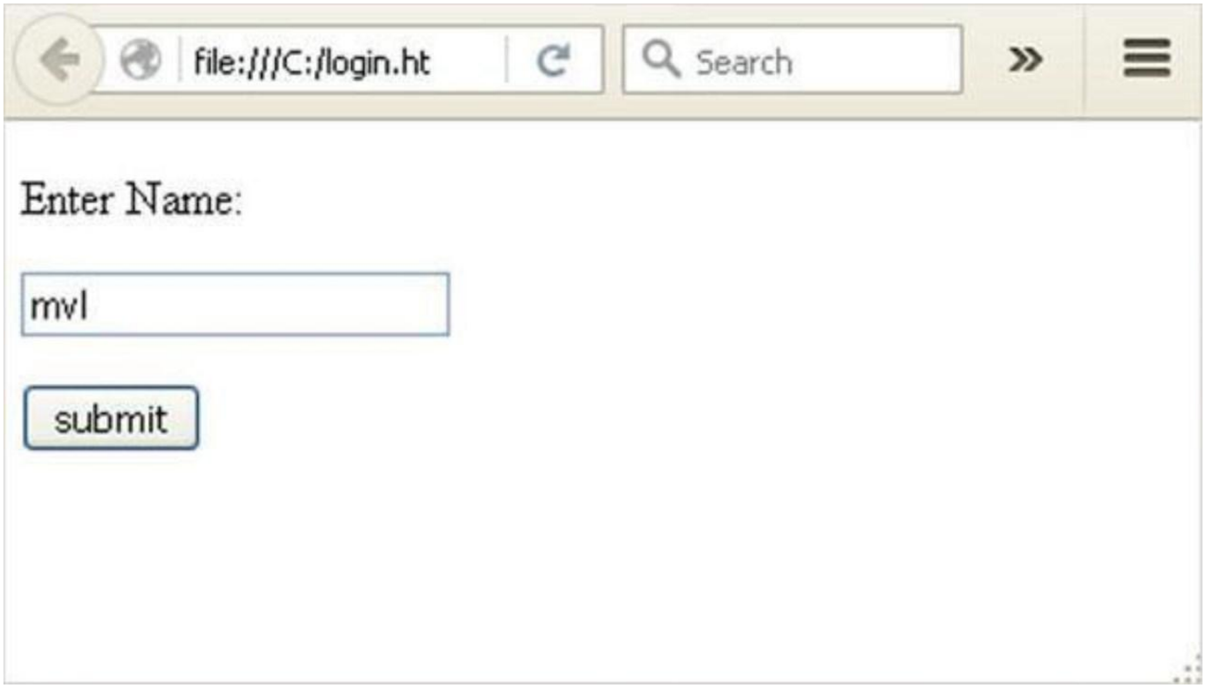


Fig 4.4: Local Host Web Server

```
from flask import Flask, redirect, url_for, request app=Flask(__name__)

@app. route('/success/<name>') def

success(name):

return 'welcome %s'% name

@app. route ('/login’, methods=['POST','GET'])

def login (): if request. method=='POST':

user=request. form['nm']



return redirect(url_for('success’, name= user)) else:
```

```
user=request.args.get('nm') return redirect (url_for

('success’, name= user)) if __name__ =='__main__':

app.run (debug =True)
```

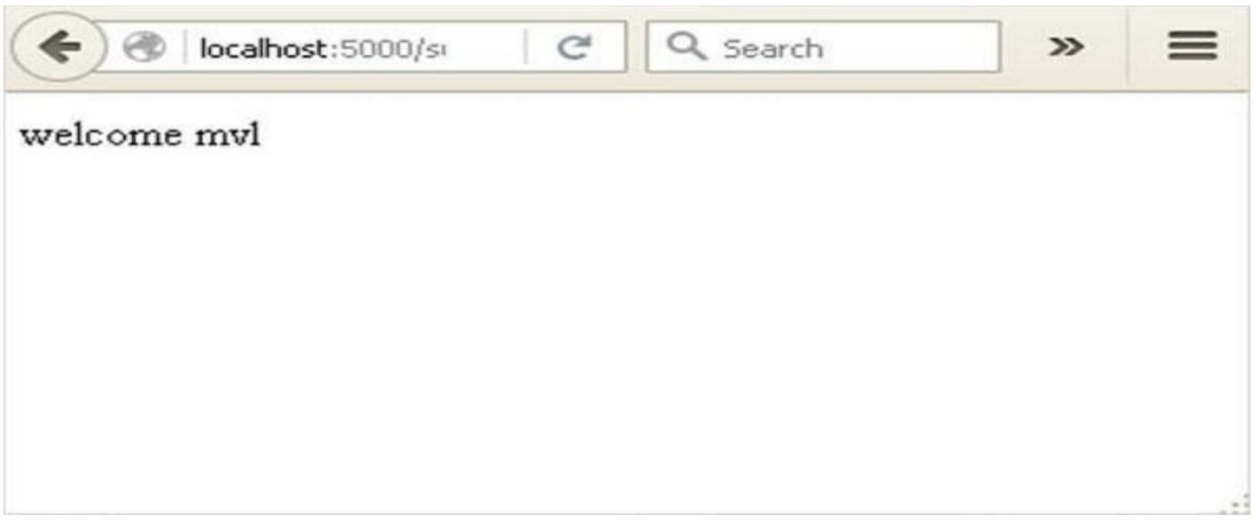
After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

Form data is Posted to the URL in action clause of form tag.

http://localhost/login is mapped to the **login ()** function. Since the server has received data by **POST** method, value of ‘nm’ parameter obtained from the form data is obtained by −

```
user = request. form['nm']
```

It is passed to **‘/success’** URL as variable part. The browser displays a **welcome** message in the window.



```
welcome mvl
```

Fig 4.5: Local Host Output Console

Change the method parameter to **‘GET’** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of ‘nm’ parameter is now obtained by −

```
User = request.args.get(‘nm’)
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to ‘nm’ parameter is passed on to ‘/success’ URL as before.

4.5 PROJECT MANAGEMENT PLAN

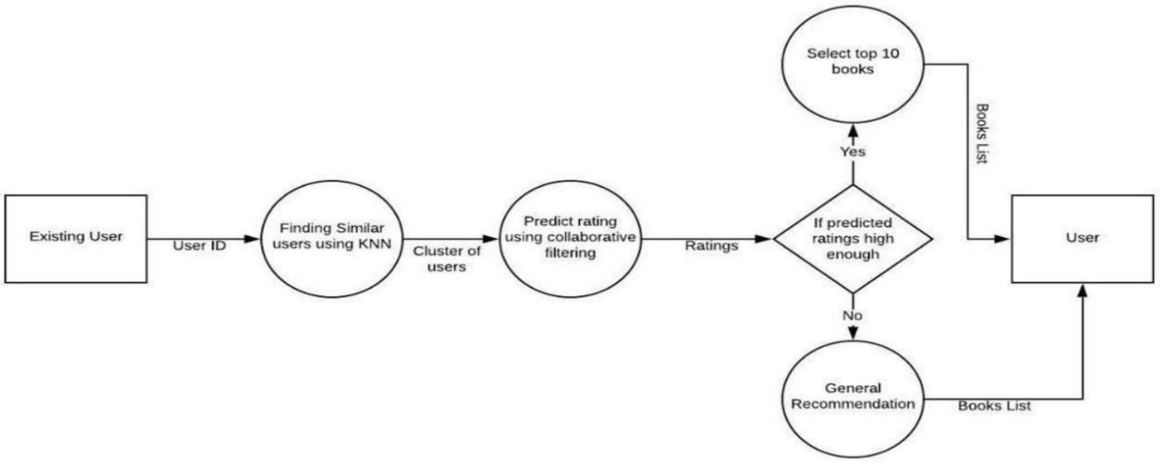


Fig 4.6: Flow Chart

CHAPTER 5

IMPLEMENTATION DETAILS

A recommendation engine is a class of machine learning which offers relevant suggestions to the customer. Before the recommendation system, the major tendency to buy was to take a suggestion from friends. But Now Google knows what news you will read, Youtube knows what type of videos you will watch based on your search history, watch history, or purchase history.

A recommendation system helps an organization to create loyal customers and build trust by them desired products and services for which they came on your site. The recommendation system today are so powerful that they can handle the new customer too who has visited the site for the first time. They recommend the products which are currently trending or highly rated and they can also recommend the products which bring maximum profit to the company.

5.1 DEVELOPMENT AND DEPLOYMENT DETAILS

Machine learning is a process that is widely used for prediction. N number of algorithms are available in various libraries which can be used for prediction. In this article, we are going to build a prediction model on historical data using different machine learning algorithms and classifiers, plot the results, and calculate the accuracy of the model on the testing data.
Building/Training a model using various algorithms on a large dataset is one part of the data. But using these models within the different applications is the second part of deploying machine learning in the real world.

To put it to use in order to predict the new data, we have to deploy it over the internet so that the outside world can use it. In this article, we will talk about how we have trained a machine learning model and created a web application on it usingFlask.

We have to install many required libraries which will be used in this model. Use pip command to install all the libraries.

pip install pandas pip install
numpy pip install sklearn

5.2 ALGORITHM

Content based Filtering: The algorithm recommends a product that is similar to those which used as watched. In simple words, In this algorithm, we try to find finding item look alike. For example, a person likes to watch Sachin Tendulkar shots, so he may like watching Ricky Ponting shots too because the two videos have similar tags and similar categories.

Collaborative based Filtering: Collaborative based filtering recommender systems are based on past interactions of users and target items. In simple words here, we try to search for the look-alike customers and offer products based on what his or her lookalike has chosen. Let us understand with an example. X and Y are two similar users and X user has watched A, B, and C movie. And Y user has watched B, C, and D movie then we will recommend A movie to Y user and D movie to X user.

Hybrid filtering method: It is basically a combination of both the above methods. It is a too complex model which recommends product based on your history as well based on similar users like you.

There are some organizations that use this method like Facebook which shows news which is important for you and for others also in your network and the same is used by Linkedin too.

Dataset description we have 3 files in our dataset which is extracted from some books selling websites.

- Books – first are about books which contain all the information related to books like an author, title, publication year, etc.
- Users – The second file contains registered user's information like user id, location.
- ratings – Ratings contain information like which user has given how much rating to which book.

So based on all these three files we can build a powerful collaborative filtering model. let's get started.

Loading data let us start while importing libraries and load datasets. while loading the file we have some problems like.

- The values in the CSV file are separated by semicolons, not by a comma.
- There are some lines which not work like we cannot import it with pandas and It throws an error because python is Interpreted language.
- Encoding of a file is in Latin

So, while loading data we have to handle these exceptions and after running the below code you will get some warning and it will show which lines have an error that we have skipped while loading.

Preprocessing Data: Now in the books file, we have some extra columns which are not required for our task like image URLs. And we will rename the columns of each file as the name of the column contains space, and uppercase letters so we will correct as to make it easy to use.

The dataset is reliable and can consider as a large dataset. we have 271360 books data and total registered users on the website are approximately 278000 and they have given near about 11 lakh rating. hence we can say that the dataset we have is nice and reliable.

We do not want to find a similarity between users or books. we want to do that If there is user A who has read and liked x and y books, And user B has also liked this two books and now user A has read and liked some z book which is not read by B so we have to recommend z book to user B. This is what collaborative filtering is.

So this is achieved using Matrix Factorization, we will create one matrix where columns will be users and indexes will be books and value will be rating. Like we have to create a Pivot table.

If we take all the books and all the users for modeling, Don’t you think will it create a problem? So what we have to do is we have to decrease the number of users and books because we cannot consider a user who has only registered on the website or has only read one or two books. On such a user, we cannot rely to recommend books to others because we have to extract knowledge from data. So what we will limit this number and we will take a user who has rated at least 200 books and also we will limit books and we will take only those books which have received at least 50 ratings from a user.

Exploratory data analysis

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**.**

Content based filtering /popularity-based filtering

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 product or movie which are in trend or are most popular among the users and directly recommend those.
In the content based collaborative filtering we are finding the highest average rating of each book. To find out the highest average rating we merge the ratings dataset with books dataset ratings_with_name=ratings.merge(books,on='ISBN')

Upon merging the rating dataset with book dataset we are grouping the ratings with respect to the name of the book
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

Using the mean function calculating the average ratings of the books.
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

By sorting the average rating we can retrive the first 50 highest rating books fro the dataset
popular_df=popular_df[popular_df['num_ratings']>=250].sort_values('avg_rating',a
scending=False).head(50) popular_df

Collaborative based filtering Approach

Collaborative filtering is used by most recommendation systems to find similar patterns or information of the users, this technique can filter out items that users like on the basis of the ratings or reactions by similar users.

Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users.

It works by searching a large group of people and finding a smaller set of users with tastes similar to a particular user. It looks at the items they like and combines them to create a ranked list of suggestions.

There are many ways to decide which users are similar and combine their choices to create a list of recommendations. This article will show you how to do that with Python.

To experiment with recommendation algorithms, you'll need data that contains a set of items and a set of users who have reacted to some of the items.

The reaction can be explicit (rating on a scale of 1 to 5, likes or dislikes) or implicit (viewing an item, adding it to a wish list, the time spent on an article).

While working with such data, you'll mostly see it in the form of a matrix consisting of the reactions given by a set of users to some items from a set of items. Each row would contain the ratings given by a user, and each column would contain the ratings received by an item.

To build a system that can automatically recommend items to users based on the preferences of other users, the first step is to find similar users or items. The second step is to predict the ratings of the items that are not yet rated by a user. 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['UserID'].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='BookRating')

Cosine Similarity

cosine similarity means the similarity between two vectors of inner product space, It is measured by the cosine of the angle between two vectors. **from sklearn.metrics.pairwise import cosine_similarity** similarity_scores=cosine_similarity(pt) similarity_scores.shape 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)[1:6]     data=[]     for i in similar_items:
      item=[]        temp_df=books[books['Book-Title']==pt.index[i[0]]]
item.extend(temp_df.drop_duplicates("Book-Title")['Book-Title'].values)
item.extend(temp_df.drop_duplicates("Book-Title")['Book-Author'].values)
item.extend(temp_df.drop_duplicates("Book-Title")['Image-URL-M'].values)
data.append(item)     return data
```

Accuracy

One of the approaches to measure the accuracy of your result is the Root Mean Square Error (RMSE), in which you predict ratings for a test dataset of user-item pairs whose rating values are already known. The difference between the known value and the predicted value would be the error. Square all the error values for the test set, find the average (or mean), and then take the square root of that average to get the RMSE.

Website Deployment

We are using the pycharm community to deploy the website. By creating the project book recommendation System.

Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality.

```
def create_app():     app =
Flask(__name__)
hello.init_app(app)     return app
```

CHAPTER 6

RESULTS AND DISCUSSION

Today the World Wide Web provides users with a vast array of information, and commercial activity on the Web has increased to the point where hundreds of new companies are adding web pages daily. This has led to the problem of information overload. Recommender systems have been developed to overcome this problem by providing recommendations that help individual users identify content of interest by using the opinions of a community of users and/or the user's preferences.

The aim of this thesis was to design and evaluate different approaches for producing personalised recommendations within the book domain. To achieve this goal, the project first investigated existing recommender systems and profiling techniques.
The next step was to build users' profiles by monitoring users' behaviour, and develop three different approaches for producing recommendations. Finally, an evaluation of the system recommendations' accuracy was done, by first conducting live user experiments and then performing offline analysis to measure the recommendations' accuracy using appropriate methods for testing.

The system evaluation results show that the accuracy of the system recommendations is very good and that a recommender system based on the combination of content-based and collaborative filtering approaches provides more accurate recommendations for the book domain.

CHAPTER 7
CONCLUSION

7.1 CONCLUSION

All of our systems– purely content-based, purely collaborative-filtering, and hybrid– performed quite well. Looking back on the project, one thing that we might have chosen to do differently in retrospect would have been to spend more time searching for a dataset of ratings with a higher rating variance per user. Had we been able to find such a dataset, our implementations of algorithms would have been tested on data that would have been more representative of what a typical commercial recommendation system could access in creating its predictions. However, given the data that was available to us, as well as the results our various approaches produced, our systems were largely successful, providing insight into how the different systems we regularly use work and the varying algorithms that make that possible.

7.2 FUTURE WORK

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.

7.3 RESEARCH ISSUES

One of the challenges faced by our research was the unavailability of reliable training datasets. In fact, this challenge faces any researcher in the field. However, although plenty of articles about predicting phishing websites using data mining techniques have been disseminated these days, no reliable training dataset has been published publically, maybe because there is no agreement in literature on the definitive features that characterize phishing websites, hence it is difficult to shape a dataset that covers all possible features.
In this article, we shed light on the important features that have proved to be sound and effective in predicting phishing websites. In addition, we proposed some new features, experimentally assign new rules to some well-known features and update some other features.

7.3 IMPLEMENTATION ISSUES

- 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 Location etc.
- Decision making on missing value imputations and outlier treatment was quite challenging as well.

REFERENCES

[1] Ahuja, Rishabh, Arun Solanki, and Anand Nayyar.” Movie recommender system using K-Means clustering and K-Nearest Neighbor.” In 2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence), pp. 263268. IEEE, 2019.

[2] Badriyah, Tessy, Erry Tri Wijayanto, Iwan Syarif, and Prima Kristalina. ”A hybrid recommendation system for E-commerce based on product description and user profile.” In 2017 Seventh International Conference on Innovative Computing Technology (INTECH), pp. 95-100. IEEE, 2017.

[3] Chen, Junnan, Courtney Miller, and Gaby G. Dagher. ”Product recommendation system for small online retailers using association rules mining.” In Proceedings of the 2014 International Conference on Innovative Design and Manufacturing (ICIDM), pp. 71-77. IEEE, 2014.

[4] Jisha, R. C., Ram Krishnan, and Varun Vikraman. ”Mobile applications recommendation based on user ratings and permissions.” In 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1000-1005.IEEE, 2018.

[5] Keerthana, N. K., Shriram K. Vasudevan, and Nalini Sampath. ”An Effective Approach to Cluster Customers with a Product Recommendation System.” Journal of Computational and Theoretical Nanoscience Vol. 17, No. 1, pp. 347-352.IEEE, 2020.

[6] Kurmashov, Nursultan, Konstantin Latuta, and Abay Nussipbekov. ”Online book recommendation System.” In 2015 Twelve International Conference on Electronics Computer and Computation (ICECCO), pp. 1-4. IEEE, 2015.

[7] Kurup, Ayswarya R., and G. P. Sajeev. ”Task recommendation in rewardbased crowdsourcing systems.” In 2017 International Conference on Advances in
Computing, Communications and Informatics (ICACCI), pp. 1511-1518. IEEE, 2017.
[8] Maya L Pai1,Suchithra M. S2andDhanya M, ”Analysis of Soil Parameters for
Proper Fertilizer Recommendation to Increase the Productivity of Paddy Field
Cultivation ”,Department of Computer Science and ITSchool of Arts and Sciences,KochiAmrita Vishwa Vidyapeetham, International Journal of Advanced Science and Technology Vol. 29, No. 03, pp. 4681-4696.IEEE, 2020

[9] M¨obius Data Scientist at Healthcare Melbourne, Victoria, Australia” Book Recommendation Dataset Build state-of-the-art models for book recommendation system”, kaggle, 2016.

[10] Mohamed, Marwa Hussien, Mohamed Helmy Khafagy, and Mohamed Hasan
Ibrahim. ” Recommender systems challenges and solutions survey.” In 2019 International Conference on Innovative Trends in Computer Engineering (ITCE), pp.
149-155. IEEE, 2019.

[11] Murali, M. Viswa, T. G. Vishnu, and Nancy Victor. ”A Collaborative Filtering based Recommender System for Suggesting New Trends in Any Domain of
Research.” In 2019 5th International Conference on Advanced Computing Communication Systems (ICACCS), pp. 550- 553. IEEE, 2019.

[12] PV Devika, K Jothisree, PV Rahul, S Arjun, Jayasree Narayan. “Book Recommendation System”, 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2021

[13] Tewari, Anand Shanker, and Kumari Priyanka. ”Book recommendation system
based on collaborative filtering and association rule mining for college students.” In 2014 International Conference on Contemporary Computing and Informatics (IC3I), pp. 135-138. IEEE, 2014.

APPENDIX

A. SOURCE CODE

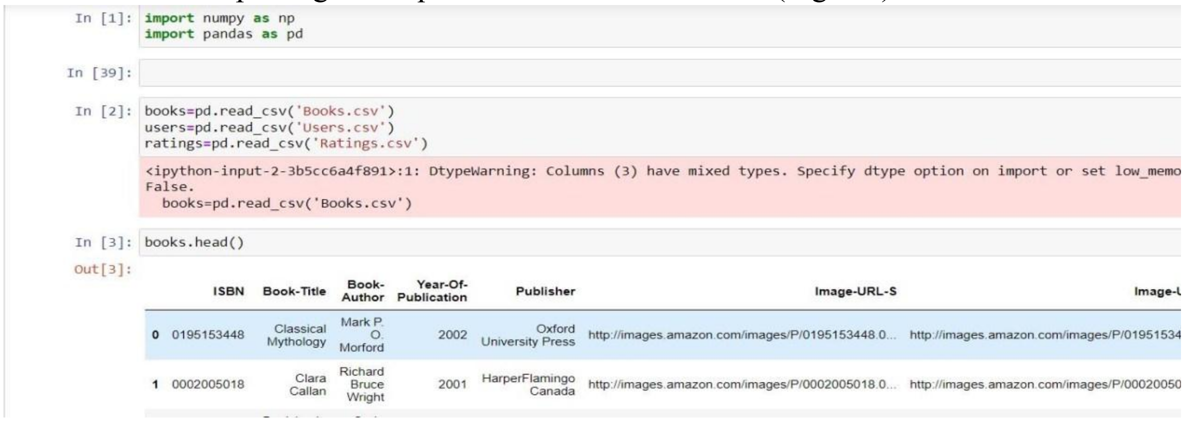
Fig A.1 Refers to the implementation of the flask for the website deployment.

```
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('index.html',
                          book_name = list(popular_df['BookTitle'].values),
                          author=list(popular_df['BookAuthor'].values),
                          image=list(popular_df['Image-URL-
M'].values),

votes=list(popular_df['num_ratings'].values),

rating=list(popular_df['avg_rating'].values)

                          )
@app.route('/recommendation') def
recommendation_ui():
    return render_template('recommendation.html')
@app.route('/recommend_books',methods=['post']) def
recommend():
    user_input=request.form.get('user_input')     index =
np.where(pt.index == user_input)[0][0]
    similar_items = sorted(list(enumerate(similarity_scores[0])), key=lambda x:
x[1], reverse=True)[1:6]     data = []     for i in similar_items:
        item = []
        temp_df = books[books['Book-Title'] == pt.index[i[0]]]
item.extend(temp_df.drop_duplicates("Book-Title")['BookTitle'].values)
        item.extend(temp_df.drop_duplicates("Book-Title")['BookAuthor'].values)
        item.extend(temp_df.drop_duplicates("Book-Title")['ImageURL-
M'].values)         data.append(item)     print(data)
    return render_template('recommendation.html',data=data)
 if __name__=='__main__':
app.run(debug=True)
```

Fig A.1: Flask Source code

B. SCREENSHOTS

First of all we are importing the required libraries and datasets (Fig B.1)

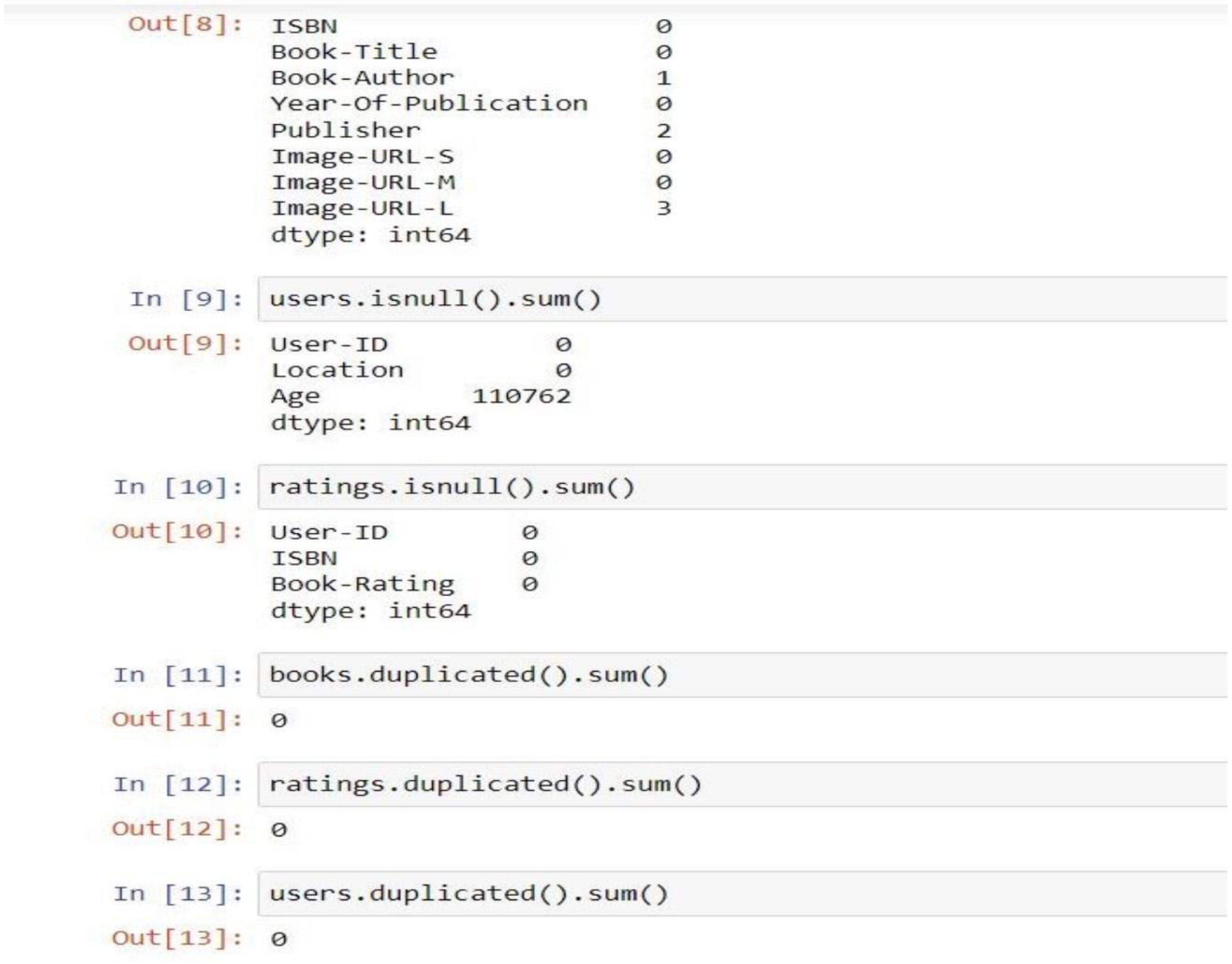


```
In [1]: import numpy as np
        import pandas as pd

In [39]:

In [2]: books=pd.read_csv('Books.csv')
        users=pd.read_csv('Users.csv')
        ratings=pd.read_csv('Ratings.csv')

        <ipython-input-2-3b5cc6a4f891>:1: DtypeWarning: Columns (3) have mixed types. Specify dtype option on import or set low_memo
        False.
          books=pd.read_csv('Books.csv')

In [3]: books.head()
Out[3]:
```

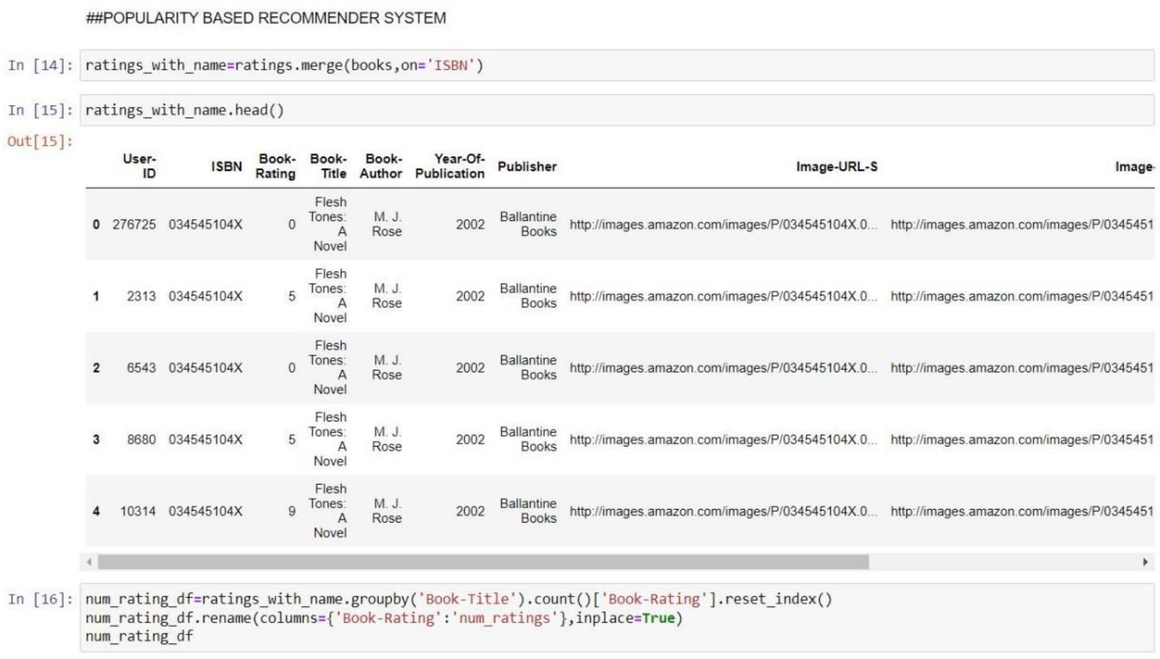
	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	Image-U
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images/P/01951534
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/00020050

Fig B.1 Importing Libraries and Datasets

In the next step data pre-processing is carried out to modify the data as required (FigB.2)



```
Out[8]:  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

In [9]:  users.isnull().sum()

Out[9]:  User-ID          0
         Location         0
         Age         110762
         dtype: int64

In [10]: ratings.isnull().sum()

Out[10]: User-ID        0
         ISBN           0
         Book-Rating    0
         dtype: int64

In [11]: books.duplicated().sum()

Out[11]: 0

In [12]: ratings.duplicated().sum()

Out[12]: 0

In [13]: users.duplicated().sum()

Out[13]: 0
```



##POPULARITY BASED RECOMMENDER SYSTEM

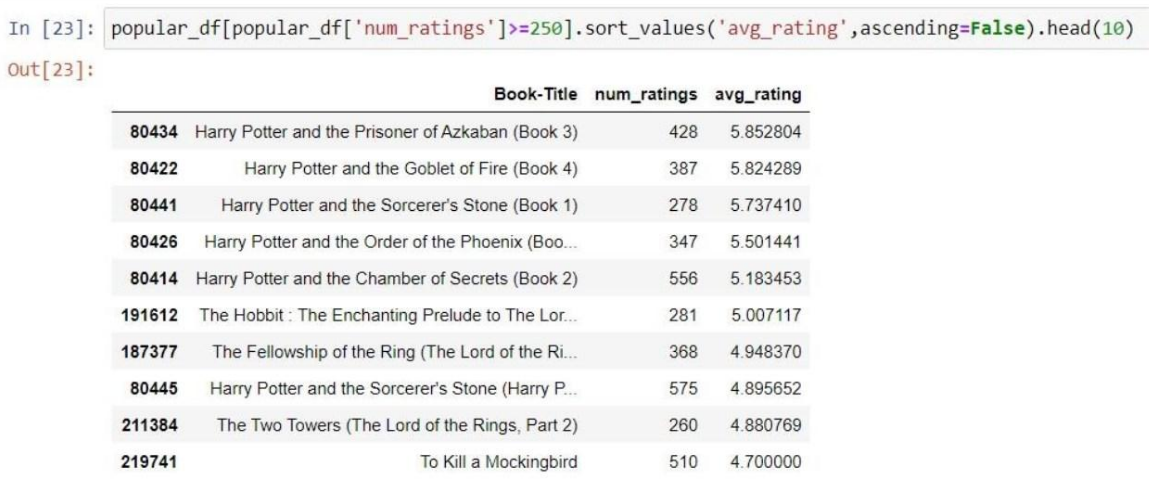
```
In [14]: ratings_with_name=ratings.merge(books,on='ISBN')

In [15]: ratings_with_name.head()

Out[15]:
```

	User-ID	ISBN	Book-Rating	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	Image-
0	276725	034545104X	0	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	http://images.amazon.com/images/P/034545104X.0...	http://images.amazon.com/images/P/0345451
1	2313	034545104X	5	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	http://images.amazon.com/images/P/034545104X.0...	http://images.amazon.com/images/P/0345451
2	6543	034545104X	0	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	http://images.amazon.com/images/P/034545104X.0...	http://images.amazon.com/images/P/0345451
3	8680	034545104X	5	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	http://images.amazon.com/images/P/034545104X.0...	http://images.amazon.com/images/P/0345451
4	10314	034545104X	9	Flesh Tones: A Novel	M. J. Rose	2002	Ballantine Books	http://images.amazon.com/images/P/034545104X.0...	http://images.amazon.com/images/P/0345451

```
In [16]: 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
```

Fig B.3 Popularity based filtering/ Content based filtering Model

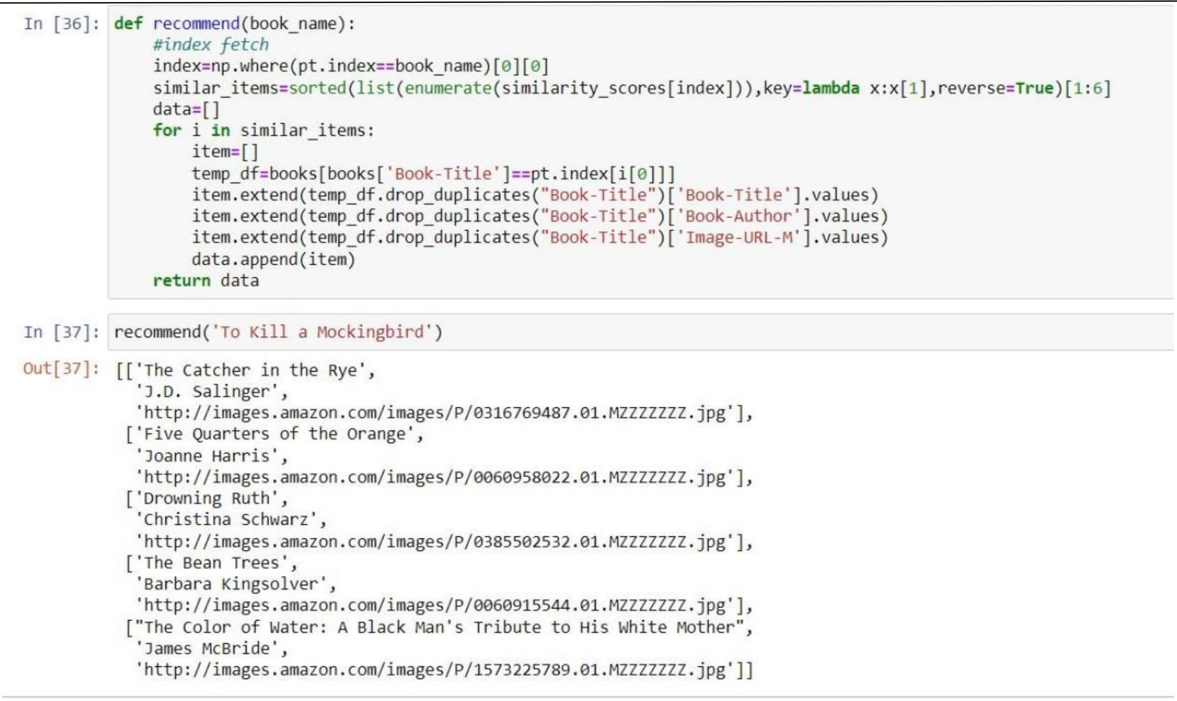
Fig B.4 describes the output of the content based filtering model i.e. the top 10 highest average rating books are displayed



```
In [23]: popular_df[popular_df['num_ratings']>=250].sort_values('avg_rating',ascending=False).head(10)

Out[23]:
```

	Book-Title	num_ratings	avg_rating
80434	Harry Potter and the Prisoner of Azkaban (Book 3)	428	5.852804
80422	Harry Potter and the Goblet of Fire (Book 4)	387	5.824289
80441	Harry Potter and the Sorcerer's Stone (Book 1)	278	5.737410
80426	Harry Potter and the Order of the Phoenix (Boo...	347	5.501441
80414	Harry Potter and the Chamber of Secrets (Book 2)	556	5.183453
191612	The Hobbit : The Enchanting Prelude to The Lor...	281	5.007117
187377	The Fellowship of the Ring (The Lord of the Ri...	368	4.948370
80445	Harry Potter and the Sorcerer's Stone (Harry P...	575	4.895652
211384	The Two Towers (The Lord of the Rings, Part 2)	260	4.880769
219741	To Kill a Mockingbird	510	4.700000

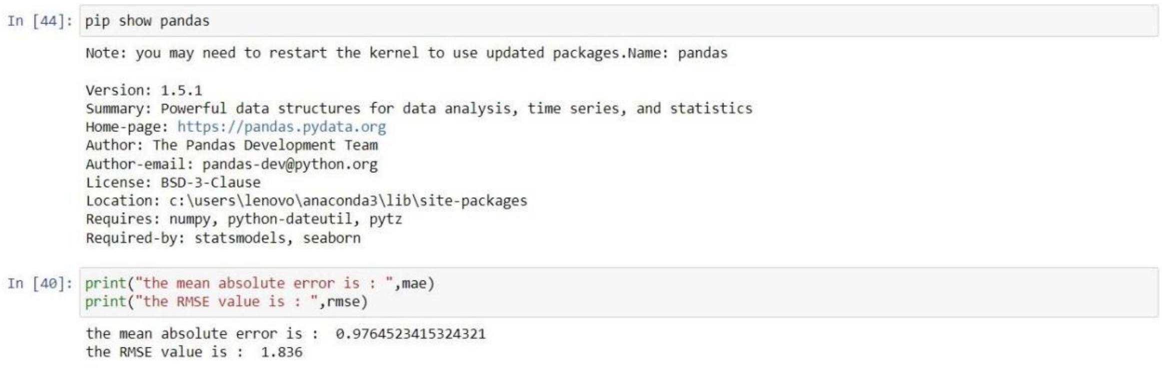
Fig B.4 Popularity based filtering / Content based filtering output Collaborative filtering model recommends the books to users based on the books interacted by the users(fig B.5)



```
In [36]: 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)[1:6]
             data=[]
             for i in similar_items:
                 item=[]
                 temp_df=books[books['Book-Title']==pt.index[i[0]]]
                 item.extend(temp_df.drop_duplicates("Book-Title")['Book-Title'].values)
                 item.extend(temp_df.drop_duplicates("Book-Title")['Book-Author'].values)
                 item.extend(temp_df.drop_duplicates("Book-Title")['Image-URL-M'].values)
                 data.append(item)
             return data

In [37]: recommend('To Kill a Mockingbird')

Out[37]: [['The Catcher in the Rye',
           'J.D. Salinger',
           'http://images.amazon.com/images/P/0316769487.01.MZZZZZZZ.jpg'],
          ['Five Quarters of the Orange',
           'Joanne Harris',
           'http://images.amazon.com/images/P/0060958022.01.MZZZZZZZ.jpg'],
          ['Drowning Ruth',
           'Christina Schwarz',
           'http://images.amazon.com/images/P/0385502532.01.MZZZZZZZ.jpg'],
          ['The Bean Trees',
           'Barbara Kingsolver',
           'http://images.amazon.com/images/P/0060915544.01.MZZZZZZZ.jpg'],
          ["The Color of Water: A Black Man's Tribute to His White Mother",
           'James McBride',
           'http://images.amazon.com/images/P/1573225789.01.MZZZZZZZ.jpg']]
```

Fig B.5 Collaborative Based Filtering model Output

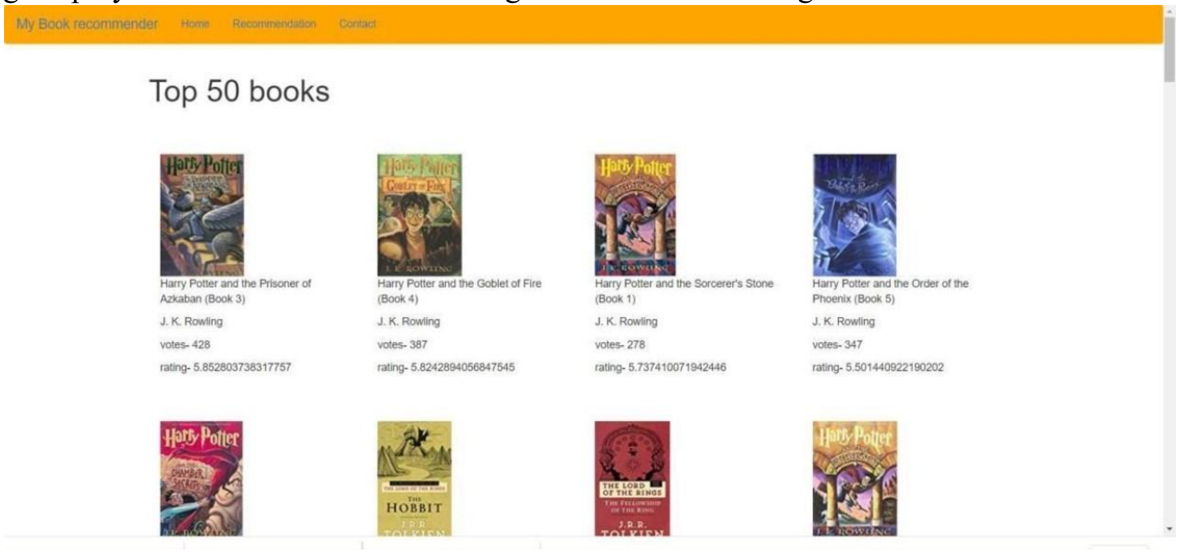
RMSE and MAE values are shown in Fig B.6



```
In [44]: pip show pandas

         Note: you may need to restart the kernel to use updated packages.Name: pandas

         Version: 1.5.1
         Summary: Powerful data structures for data analysis, time series, and statistics
         Home-page: https://pandas.pydata.org
         Author: The Pandas Development Team
         Author-email: pandas-dev@python.org
         License: BSD-3-Clause
         Location: c:\users\lenovo\anaconda3\lib\site-packages
         Requires: numpy, python-dateutil, pytz
         Required-by: statsmodels, seaborn

In [40]: print("the mean absolute error is : ",mae)
         print("the RMSE value is : ",rmse)

         the mean absolute error is :  0.9764523415324321
         the RMSE value is :  1.836
```

Fig B.6 RMSE and MAE valuesw

Webpage deployment of content based filtering model is shown in Fig B.7



My Book recommender Home Recommendation Contact

Top 50 books

Harry Potter and the Prisoner of Azkaban (Book 3)
J. K. Rowling
votes- 428
rating- 5.852803738317757

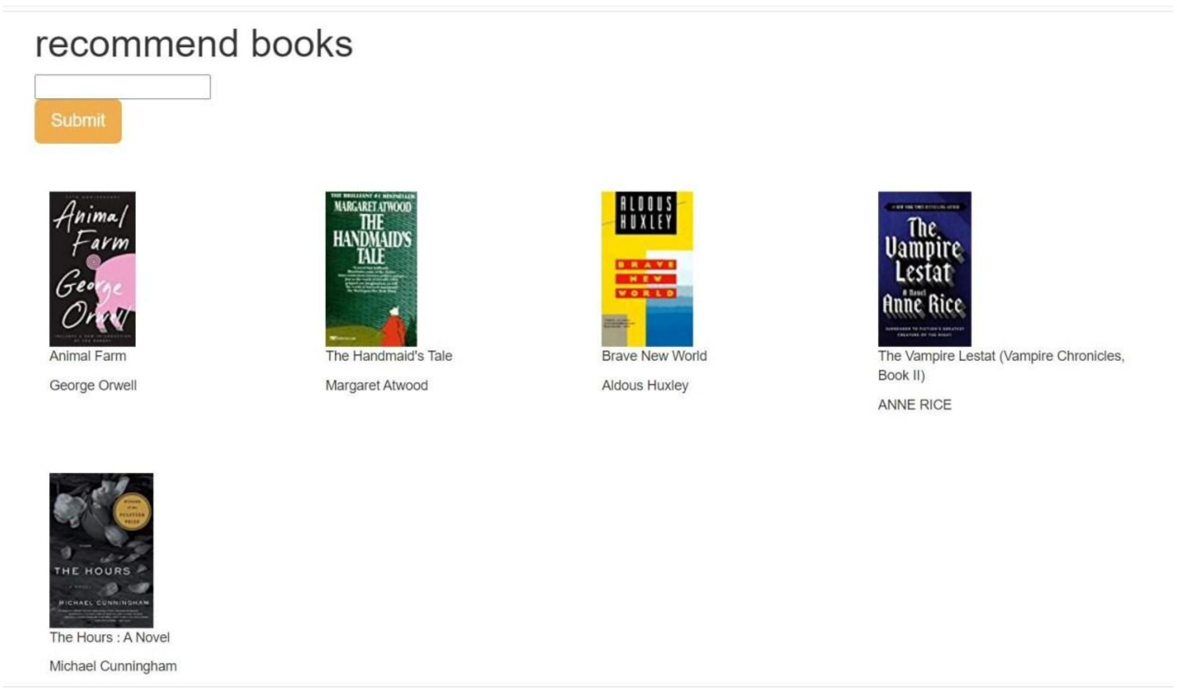
Harry Potter and the Goblet of Fire (Book 4)
J. K. Rowling
votes- 387
rating- 5.8242894056847545

Harry Potter and the Sorcerer's Stone (Book 1)
J. K. Rowling
votes- 278
rating- 5.737410071942446

Harry Potter and the Order of the Phoenix (Book 5)
J. K. Rowling
votes- 347
rating- 5.501440922190202

Fig B.7 Webpage for content-based filtering Model

Webpage deployment of collaborative based filtering is shown in the fig B.8



Animal Farm
George Orwell

The Handmaid's Tale
Margaret Atwood

Brave New World
Aldous Huxley

The Vampire Lestat (Vampire Chronicles, Book II)
ANNE RICE

The Hours : A Novel
Michael Cunningham

Fig B.8 Webpage for collaborative based filtering model