

**BOOK RECOMMENDER SYSTEM
USING ARTIFICIAL INTELLIGENCE
A COURSE PROJECT REPORT**

By

**Shivendra Bharuka (RA2011031010020)
Aaryan Prasad (RA2011031010040)
Dimple Singhvi (RA2011031010046)**

Under the guidance of
Dr. V.R. Balasaraswathi
In partial fulfillment for the Course
of

18CSC305J - AI
in Computer Science (Information Technology)



FACULTY OF ENGINEERING AND TECHNOLOGY

**SRM INSTITUTE OF SCIENCE AND
TECHNOLOGY**

Kattankulathur, Chengalpattu District

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report of "**Book recommender system using artificial intelligence**" is the Bonafide work of **Shivendra Bharuka (RA2011031010020)**, **Aaryan Prasad (RA2011031010040)** and **Dimple Singhvi (RA2011031010046)** who carried out the project work under my supervision.

Date of review - 24/04/23

SIGNATURE

Dr. V.R. Balasaraswathi,
Assistant Professor,
Networking and Communication,
SRM Institute of Science and Technology
Potheri, SRM Nagar,
Kattankulathur, Tamil Nadu 603203

TABLE OF CONTENTS

1. INTRODUCTION

- 1.1. Introduction
- 1.2. Introduction to Machine Learning
 - 1.2.1. Supervised learning
 - 1.2.2. Unsupervised learning
 - 1.2.3. Applications of Machine Learning
- 1.3. Clustering
- 1.4. Motivation of the work
- 1.5. Problem Statement
- 1.6. Organization of Thesis

2. LITERATURE SURVEY

- 2.1. Collaborative Filtering with Jaccard Similarity to build a recommendation system
- 2.2. Building a Recommendation System using Keras Deep learning Framework
- 2.3. Using Quick sort Algorithm approach to design a system
- 2.4. Using UV Decomposition and KNN for building system
- 2.5. Recommending books through CB and CF approaches
- 2.6. Detecting patterns, correlations and uses Collaborative Filtering and Associative Rule Mining
- 2.7. Uses Demographic, Collaborative Filtering, Content-based to build a Hybrid Recommender System
- 2.8. PHP-based CF, Fuzzy logic, Context Engine for recommendation
- 2.9. Hybrid Recommender System through Collaborative Filtering
- 2.10. Uses a Machine Learning algorithm to build a system

3. PROPOSED WORK

- 3.1. Proposed System
 - 3.1.1. System Architecture
- 3.2. Modules Division
 - 3.2.1. Data Acquisition
 - 3.2.2. Data Preprocessing
 - 3.2.3. Feature Extraction
 - 3.2.4. Training Methods
 - 3.2.4.1. K-Means Clustering
 - 3.2.4.2. Gaussian Mixture
 - 3.2.5. Testing Data
 - 3.2.6. Content Based Filtering
 - 3.2.7. User Interface

4. IMPLEMENTATION

- 4.1. System Requirements
 - 4.1.1. Functional Requirements
 - 4.1.2. Non-Functional Requirements
- 4.2. System Configuration
 - 4.2.1. Software Requirements
 - 4.2.1.1. Introduction to Python
 - 4.2.1.2. Introduction to Django Framework
 - 4.2.1.3. Python Libraries
 - 4.2.2. Hardware Requirements
- 4.3. Feasibility Study
 - 4.3.1. Economic Feasibility
 - 4.3.2. Technical Feasibility
 - 4.3.3. Operational Feasibility
- 4.4. Sample Code

4.5. Experimental analysis and Performance Measures

4.5.1. Performance Analysis and models Comparison

5. RESULT

6. CONCLUSION

7. FUTURE ENHANCEMENT

8. REFERENCES

INTRODUCTION

Now-a-days, online rating and reviews are playing an important role in books sales. Readers were buying books depend on the reviews and ratings by the others. Recommender system focuses on the reviews and ratings by the others and filters books. In this paper, Hybrid recommender system is used to boost our recommendations. The technique used by recommender systems is Collaborative filtering. This technique filters information by collecting data from other users. Collaborative filtering systems apply the similarity index-based technique. The ratings of those items by the users who have rated both items determine the similarity of the items. The similarity of users is determined by the similarity of the ratings given by the users to an item. Content-based filtering uses the description of the items and gives recommendations which are like the description of the items. With these two filtering systems, books are recommended not only based on the user's behavior but also with the content of the books. So, our recommendation system recommends books to the new users also. In this recommender system, books are recommended based on collaborative filtering technique and similar books are shown using content-based filtering. The required dataset for the training and testing of our model is downloaded from Good-Reads website. Matrix Factorization technique such as Truncated-SVD which takes sparse matrix of dataset is used for reduction of features. The reduced dataset is used for clustering to build a recommendation system. Clustering is a collaborative filtering technique that is used to build our recommendation system in which data points are grouped into clusters. In this paper, we used two methods i.e., K-means and Gaussian mixture for clustering the users. The better model is selected based on the silhouette score and used for clustering. Silhouette score or silhouette coefficient is used to calculate how good the clustering is done. Negative value shows that clustering is imperfect whereas positive value shows that clustering was done perfectly. Difference between the mean rating before clustering and after clustering is calculated. Root Mean square Error is used to measure the error between the absolute 2 values and obtained values. That RMSE value is used to find the fundamental accuracy.

1.2. Introduction to Machine Learning:

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. It gives the computer that makes it more like humans i.e., ability to learn. Machine learning is used in many streams than anyone would accept. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves. The process of learning begins with observations or data, such as examples, direct experience, or instruction, to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly. Machine Learning is a sub-area of artificial intelligence, whereby the term refers to the ability of IT systems to independently find solutions to problems by recognizing patterns in databases. In other words: Machine Learning enables IT systems to recognize patterns based on existing algorithms and data sets and to develop adequate solution concepts. Therefore, in Machine Learning, artificial knowledge is generated based on experience. To enable the software to independently generate solutions, the prior action of people is necessary. For example, the required algorithms and data must be fed into the systems in advance and the respective analysis rules for the recognition of patterns in the data stock must be defined. Once these two steps have been completed, the system can perform the following tasks by Machine Learning:

- Finding, extracting and summarizing relevant data
- Making predictions based on the analysis data
- Calculating probabilities for specific results

Basically, algorithms play an important role in Machine Learning: On the one hand, they are responsible for recognizing patterns and on the other hand, they can generate solutions. Algorithms can be divided into different categories:

1.2.1. Supervised learning:

During monitored learning, example models are defined in advance. To ensure an adequate allocation of the information to the respective model groups of the algorithms, these then must be specified. In other words, the system learns based on given input and output pairs. During monitored learning, a programmer, who acts as a kind of teacher, provides the appropriate values for a particular input. The aim is to train the system in the context of successive calculations with different inputs and outputs to establish connections. Supervised learning is where you have input variables (X) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output. $Y = f(X)$ The goal is to approximate the mapping function so well that when you have new input data (X) that you can predict the output variables (Y) for that data. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

Techniques of Supervised Machine Learning algorithms include linear and logistic regression, multi-class classification, Decision Tree and Support Vector Machine. Supervised Learning problems can be further grouped into Regression and Classification problems. The difference between these two is that the dependent attribute is numerical for regression and categorical for classification:

- **Regression:**

Linear regression is a linear model, e.g., a model that assumes a linear relationship between the input variables (x) and the single output variable (y). More specifically, that y can be calculated from a linear combination of the input variables (x). When there is a single input variable (x), the method is referred to as simple linear regression. When there are multiple input variables, literature from statistics often refers to the method as multiple linear regression.

- **Classification:**

Classification is a process of categorizing a given set of data into classes, it can be performed on

both structured and unstructured data. The process starts with predicting the class of given data points. The classes are often referred to as target, label or categories. In short classification either predicts categorical class labels or classification data based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are number of classification models. Classification models include Logistic Regression, Decision Tree, Random Forest, Gradient Boosted Tree, One-vs.-One and Naïve Bayes.

1.2.2. Unsupervised learning:

In unsupervised learning, artificial intelligence learns without predefined target values and without rewards. It is mainly used for learning segmentation (clustering). The machine tries to structure and sort the data entered according to certain characteristics. For example, a machine could (very simply) learn that coins of different colors can be sorted according to the characteristic "color" to structure them. Unsupervised Machine Learning algorithms are used when the information used to train is neither classified nor labeled. The system does not figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data. Unsupervised Learning is the training of Machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Unsupervised Learning is classified into two categories of algorithms:

- **Clustering:**

A clustering problem is where you want to discover the inherent grouping in the data such as grouping customers by purchasing behavior.

- **Association:**

An Association rule learning problem is where you want to discover rules that describe large portions of your data such as people that buy X also tend to buy Y.

1.2.3. Applications of Machine Learning:

Virtual Personal Assistants: Siri, Alexa, Google Now are some of the popular examples of virtual

personal assistants. As the name suggests, they assist in finding information, when asked over voice. Machine learning is an important part of these personal assistants as they collect and refine the information based on your previous involvement with them. Later, this set of data is utilized to render results that are tailored to your preferences.

Virtual Assistants are integrated to a variety of platforms. For example:

- Smart Speakers: Amazon Echo and Google Home
- Smartphone: Samsung Bixby on Samsung S8
- Mobile Apps: Google Allo

Video Surveillance:

- Imagine a single person monitoring multiple video cameras! Certainly, a difficult job to do and boring as well. This is why the idea of training computers to do this job makes sense.
- The video surveillance system nowadays is powered by AI that makes it possible to detect crime before they happen. They track unusual behaviors of people like standing motionless for a long time, stumbling, or napping on benches etc. The system can thus give an alert to human attendants, which can ultimately help to avoid mishaps. And when such activities are reported and counted to be true, they help to improve the surveillance services. This happens with machine learning doing its job at the backend.

Social Media Services:

From personalizing your news feed to better ads targeting, social media platforms are utilizing machine learning for their own and user benefits.

- People You May Know
- Face Recognition

Search Engine Result Refining:

Google and other search engines use machine learning to improve the search results for you. Every time you execute a search, the algorithms at the backend keep a watch at how you respond to the results. If you open the top results and stay on the web page for long, the search engine assumes that the results it displayed were in accordance to the query. Similarly, if you reach the second or third page of the search results but do not open any of the results, the search engine estimates that the results served did not match requirement. This way, the algorithms working at the backend improve the search results.

1.3. Clustering

Clustering is an unsupervised learning method in which we draw references from datasets consisting of input data without labelled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent.

Clustering is the task of dividing the population or data points into several groups such that data points in the same groups are more like other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects based on similarity and dissimilarity between them. Clustering is very important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for good clustering. It depends on the user, what is the criteria they may use which satisfy their need. This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

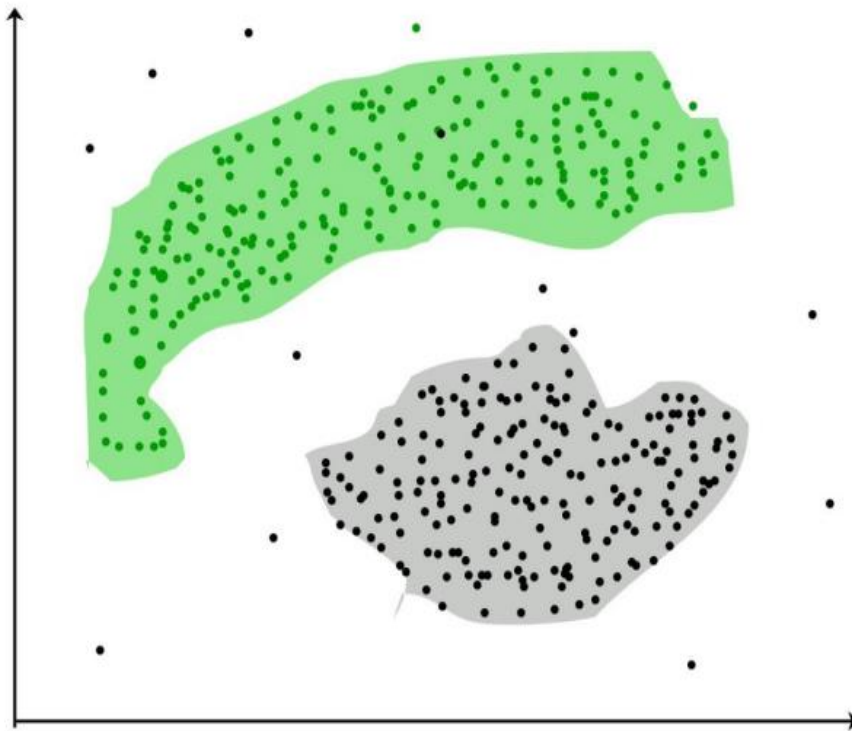


Fig 1.1 Clustering of data points

Clustering Methods:

- **Density-Based Methods:** These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters. Example: DBSCAN (Density-Based Spatial Clustering of Applications with Noise), OPTICS (Ordering Points to Identify Clustering Structure) etc.
- **Hierarchical Based Methods:** The clusters formed in this method forms a tree-type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two categories:
 - Agglomerative (bottom-up approach)
 - Divisive (top-down approach)

Examples: CURE (Clustering Using Representatives), BIRCH (Balanced Iterative Reducing Clustering and using Hierarchies) etc.

- **Partitioning Methods:** These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter 8 example K-means, CLARANS (Clustering Large Applications based upon Randomized Search) etc.
- **Grid-based Methods:** In this method the data space is formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (Clustering in Quest) etc.

In this paper, partitioning method of clustering is used. We used Clustering algorithm which is simplest unsupervised learning algorithm in this paper, and it partition n observations into k clusters where each observation belongs to the cluster.

1.4. Motivation of the work

In the present world, all products are buying based on the reviews and ratings by the others.

There are so many products with high rating and reviews, but we only put our interest in some products which we like. Recommender system works on this principle only i.e., it recommends products based on the interest of the users. Our idea is to create recommender system that recommends books based on the user's interest i.e., we recommend books which are like the books that user already liked. It can also recommend books which are liked by similar users. Similar users are those who liked the books which are liked by the current user.

We also add another feature i.e., we recommend books which are independent of the user's interest. With this feature, we can recommend books to the new users also. Book recommendation sites that were available online now a days shows the books which are recommended by the system. Here, we are also recommending books based on the description of the book. We will get books which are like the book we selected in this system. For that purpose, only, we built Hybrid recommender system. Hybrid recommender system is a combination of Collaborative Filtering system and Content Based Filtering system.

1.5. Problem Statement

Recommending books using Machine learning algorithm is the main goal of this project. Books are recommended by the clustering model, and we are going to train and build using various features such as user's rating, book description, book titles etc. The system groups users into clusters so that each data point within cluster is similar and dissimilar to the data point in the other cluster. The system we would like to develop will also be able to find an average rating for each cluster and it is going to find top rated books of users from each cluster. All these books shortlisted by our system will be used for training our model in future. The prediction model needs to be trained to produce better results.

1.6. Organization of Thesis

The chapters of this document describe the following:

Chapter-1 is about the introduction of our project where we have given clear insights about our project domain and other related concepts.

Chapter-2 specifies about literature survey where all different existing methods and models are examined.

Chapter-3 specifies about proposed system with a system architecture along with detailed explanations of each module.

Chapter-4 specifies about the implementation of our system along with performance measures and comparisons between different models. It also specifies about implementation along with sample code. Chapter-5 gives the conclusion to our work with an insight for the future scope.

Chapter-5 includes result

Chapter-6 gives the conclusion to our work

Chapter-7 gives an insight for the future scope

CHAPTER 2 – LITERATURE SURVEY

Most researchers used Pearson's Correlation Coefficient function to calculate similarity among book ratings to recommend books.

2.1 Collaborative Filtering with Jaccard Similarity to build a recommendation system

Avi Rana and K. Deeba, et.al. (2019) [1] proposed a paper "Online Book Recommendation System using Collaborative Filtering (With Jaccard Similarity)". In this paper, the author used CF with Jaccard similarity to get more accurate recommendations because general CF difficulties are scalability, sparsity, and cold start. So, to overcome these difficulties, they used CF with Jaccard Similarity. JS is based on pair of books index which is a ratio of common users who have rated both books divided by the sum of users who have rated books individually. Books with a high JS index are highly recommended.

2.2 Building a Recommendation System using Keras Deep learning Framework

G. Naveen Kishore, et.al. (2019) [2] proposed a paper "Online Book Recommendation System". The dataset used in this paper was taken from the website "good books-10k dataset" which contains ten thousand unique books. Features are book_id, user_id, and rating. In this paper, the author adopted a Keras deep learning framework model to create neural network embedding.

2.3 Using Quick sort Algorithm approach to design a system

Uko E Okon, et.al. (2018) [3] proposed a paper "An Improved Online Book Recommender System using Collaborative Filtering Algorithm". The authors designed and developed a recommendation model by using a quick sort algorithm, collaborative filtering, and object-oriented analysis and design methodology (OOADM). This system produces an accuracy of 90-95%.

2.4 Using UV Decomposition and KNN for building system

Jinny Cho, et.al. (2016) [4] proposed a paper “Book Recommendation System”. In this paper, the author uses two approach methods which are Content based (CB) and Collaborative Filtering (CF). They used two algorithms as UV-Decomposition and K Nearest Neighbors (KNN). They obtained a result with an accuracy of 85%.

2.5 Recommending books through CB and CF approaches

Sushma Rjpurkar, et.al. (2015) [5] proposed a paper “Book Recommendation System”. In this paper, the author used Associative Rule Mining to find association and correlation relationships among a dataset of items. They used CB and CF approaches to build a system.

2.6 Detecting patterns, correlations and uses Collaborative Filtering and Associative Rule Mining

Abhay E. Patil, et.al. (2019) [6] proposed a paper “Online Book Recommendation System using Association Rule Mining and Collaborative Filtering”. The author detected recurrently occurring patterns, correlations and uses various databases such as relational databases, transactional databases to form associations. They used two approaches i.e., User-based and Item-based Collaborative Filtering, and used the Pearson correlation coefficient to find similarity between the items.

2.7 Uses Demographic, Collaborative Filtering, Content-based to build a Hybrid Recommender System

Suhas Patil, et.al. (2016) [7] proposed a paper “A Proposed Hybrid Book Recommender System”. In this paper, the author used techniques such as 14 Demographic, Collaborative Filtering, Content-based to build a system and rarely they combined the features of these techniques to

make a better recommendation system.

2.8 PHP-based CF, Fuggy logic, Context Engine for recommendation systems

Ankit Khera, et.al. (2008) [8] proposed a paper “Online Recommendation System”. In this paper, the author used the User similarity matrix, Vogoo which is PHP-based CF, Fuggy logic, Context Engine for building recommendation systems. Pearson Correlation is a similarity function in this paper.

2.9 Hybrid Recommender System through Collaborative Filtering

Anagha Vaidya and Dr. Subhash Shinde, et.al. (2019) [9] proposed a paper “Hybrid Book Recommendation System”. In this paper, the author used techniques such as Collaborative Filtering etc. and used the Pearson correlation coefficient. It was published in International Research Journal of Engineering and Technology (IRJET).

2.10 Using Machine Learning Algorithm to build a system

Dhirman Sarma, Tanni Mittra and Mohammad Shahadat Hossain, et.al. (2019) [10] proposed a paper “Personalized Book Recommendation System using Machine Learning Algorithm”. It was published in The Science and Information Organization vol.12.

CHAPTER 3 – PROPOSED WORK

3.1 Proposed system

3.1.1 System Architecture

System Architecture describes “the overall structure of the system and the ways in which the structure provides conceptual integrity”. The system architecture to build a recommendation system involves the following five major steps.

3.2.1 Data Acquisition

3.2.2 Data Pre-processing

3.2.3 Feature Extraction

3.2.4 Training Methods

3.2.5 Testing Data

In Step 3.2.1, Dataset was collected from Good Reads Website in which three datasets are present i.e., Books Dataset, Ratings Dataset, Users Dataset. In Step 3.2.2, Datasets were pre-processed to make suitable for developing the Recommendation system. In Step 3.2.3, Feature extraction is performed in which Truncated-SVD is used to reduce the features of the dataset and Data splitting is done in which training dataset and testing dataset are divided into 80:20 ratio. In Step 3.2.4, Content Based Filtering System is developed in which book description is taken as an input and Collaborative Filtering System is developed by building a model using K-Means Algorithm over Gaussian Mixture after comparing with Silhouette scores. In step 3.2.5, Testing of model with test data is performed.

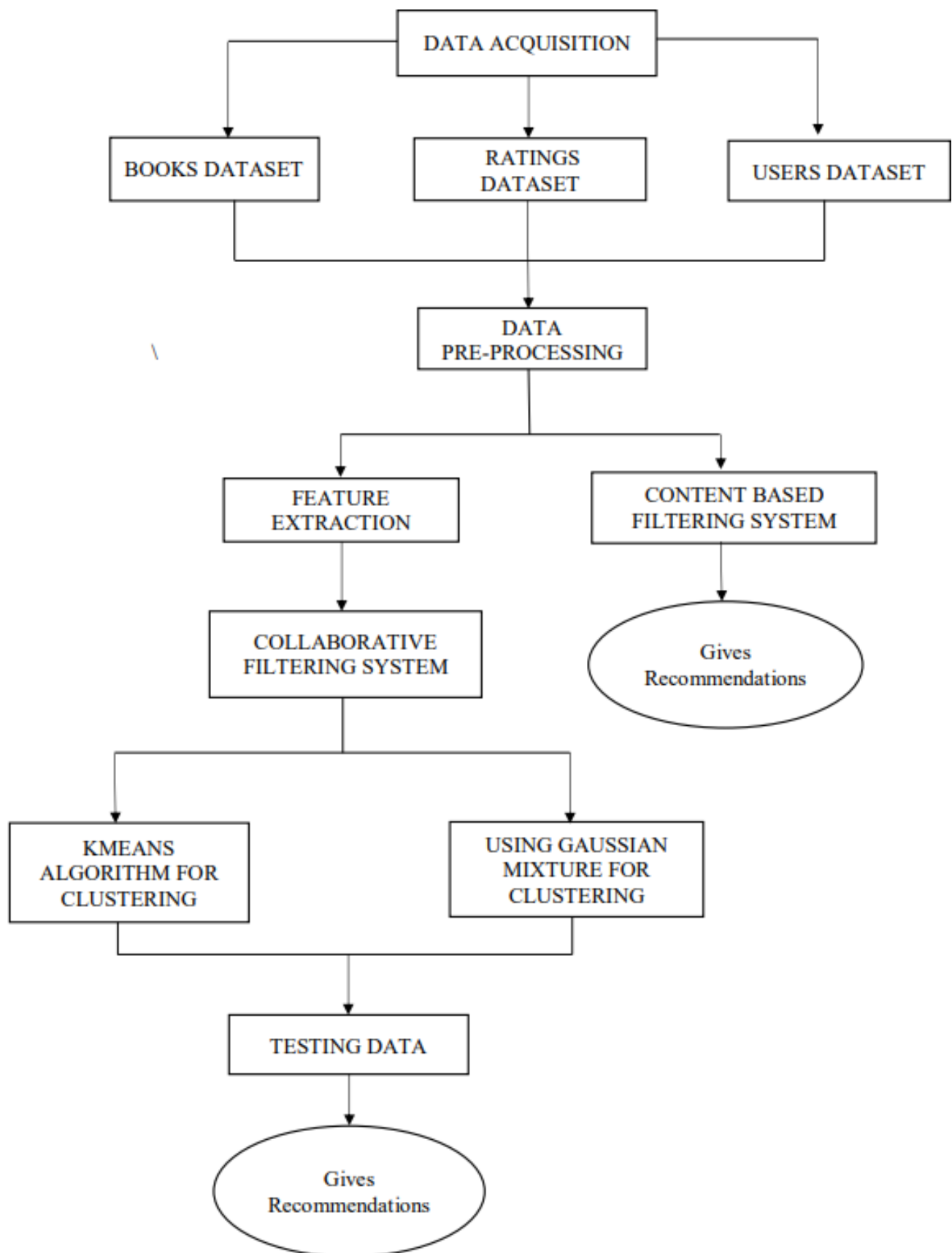


Fig-3.1 System Architecture

3.2 Modules Division

Let us discuss about the various modules in our proposed system and what each module contributes to achieving our goal.

3.2.1 Data Acquisition:

The goal of this step is to find and acquire all the related datasets or data sources. In this step, the main aim is to identify various available data sources, as data are often collected from various online sources like databases and files. The size and the quality of the data in the collected dataset will determine the efficiency of the model. The Books dataset is collected from the Goodreads website.

3.2.2 Data Pre-Processing:

The goal of this step is to study and understand the nature of data that was acquired in the previous step and to know the quality of data. In this step, we will check for any null values and remove them as they may affect the efficiency. Identifying duplicates in the dataset and removing them is also done in this step.

3.2.3 Feature Extraction:

After pre-processing the acquired data, the next step is to reduce the features i.e. Dimensionality reduction. The reduced features should be able to give high efficiency. We used Matrix Factorization technique such as Truncated SVD which takes sparse matrix as input for reduction of features.

Splitting the Dataset into the Training set and Test set:

In machine learning projects, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model

by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

Usually, dataset will be split into train and test in the ratio of 8:2 i.e., 80 percent of data is used for training and 20 percent of data is used for testing the model. We have also done in the same way. It can be seen in the above Fig 3.10.

3.2.4 Training Methods:

Now, we have our training and testing data. The next step is to identify the possible training methods and train our models. We have used two different clustering methods for training models. After that based on the silhouette score of each model, we would decide on which model to use finally.

3.2.4.1 K-Means Clustering: K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters in such a manner that each dataset belongs to only one group that has similar features. Here K defines the number of pre-defined clusters. We must associate each cluster with a centroid in this algorithm. The sum of distances between the data point and their corresponding clusters should be minimized. The unlabeled dataset is taken as an input and the dataset into k-number of clusters is divided, and the process is repeated until it does not find the best clusters. We must predetermine the k value in this algorithm. Elbow method is used to find the value of k which decides the number of clusters. This method uses the Within Cluster Sum of Squares (WCSS) value that defines the total variations within a cluster.

The Formula for calculating the value of WCSS for n clusters is as follows:

$$WCSS = \sum_{P_i \text{ in Cluster } 1} \text{distance}(P_i, C_1)^2 + \sum_{P_i \text{ in Cluster } 2} \text{distance}(P_i, C_2)^2 + \dots + \sum_{P_i \text{ in Cluster } n} \text{distance}(P_i, C_n)^2$$

The basic steps involved in K-Means Clustering algorithm is as follows:

Step-1: Select the number K which gives the number of clusters.

Step-2: Select random K number of points or centroids.

Step-3: Each data point to their nearest centroid should be assigned, which forms the predefined K clusters.

Step-4: Calculate the variance and place a new centroid for each cluster.

Step-5: We have to repeat the step-3, each data-point to the new closest centroid of each cluster should be reassigned.

Step-6: If reassignment happens, then go to step-4 or else go to step-7.

Step-7: Stop.

In scikit-learn python library, `sklearn.cluster.KMeans` module is used for carrying out K-Means Clustering. We must specify the number of clusters a parameter for this function. We will use our training dataset to fit the model. Fig 3.11 shows the sample code for training model using KMeans.

3.2.4.2 Gaussian Mixture:

Gaussian Mixture models are powerful clustering algorithms. It assumes that there are a certain number of Gaussian distributions where each distribution represents a cluster. This model groups the data points together into a single distribution. These models used the soft clustering technique for assigning data points to Gaussian distributions.

In a one-dimensional space, the probability density function of a Gaussian distribution (univariate) is as follows:

$$P(x | \mu, \sigma^2) = N(x; \mu, \sigma^2) = 1/Z [\exp(-(x - \mu)^2 / 2\sigma^2)]$$

Where,

Z is the normalization constant i.e., $Z = \sqrt{2\pi\sigma^2}$,

μ is the mean i.e., $\mu = E[x]$, and

σ^2 is the variance of the distribution i.e., $\sigma^2 = \text{var}[x]$.

In a multidimensional space, the probability density function of a Gaussian distribution (multivariate) is as follows:

$$P(x | \mu, \Sigma) = N(x; \mu, \Sigma) = 1/Z [\exp(-1/2 (x-\mu)^T \Sigma^{-1} (x-\mu))]$$

Where,

X is the input vector,

μ is the 2D mean vector, and

Σ is the 2×2 covariance matrix

Thus, we would have K (number of clusters) Gaussian distributions.

3.2.5 Testing Data

Once Clustering model has been trained on pre-processed dataset, then the model is tested using different data points. In this testing step, the model is checked for the silhouette score for checking goodness of clustering. All the training methods need to be verified for finding out the best model to be used. In figures 3.10, 3.11, after fitting our model with training data, we used this model to predict values for test dataset. These predicted values on testing data are used for models' comparison. The users in the test set, on average, rated their clusters' favorite books higher than a random set of 10 books by 0.47 stars, or nearly half a star.

3.3 Content Based Filtering

A Content-Based filtering system recommends items that are like the content of the item. This System uses the description of the items and gives the recommendations that are like the description. We used Cosine similarity as a similarity function for this system. The Item-Content Matrix which describes the attributes of the features is taken as an input. Based on the angle between the vectors, Cosine similarity is calculated. We improve the quality of the content-based system by normalizing and tuning the attributes with the use of the TF-IDF vectorizer. TF (Term Frequency) is termed as a word frequency in a document. IDF (Inverse Document Frequency) is universe document frequency. The TF-IDF Vectorizer will tokenize documents, learn the vocabulary and inverse document frequency weightings, and allow you to encode new documents. It transforms text to feature vectors that can be used as input to estimator. Vocabulary is a dictionary that converts each token (word) to feature index in the matrix, each unique token gets a feature index. It tells you that the token 'me' is represented as feature number 8 in the output

matrix. A vocabulary of 8 words is learned from the documents and each word is assigned a unique integer index in the output vector. In this paper, TF-IDF that takes stop_words as a parameter transforms book description into matrix of vectors.

3.4 User Interface

User interface is very essential for any project because everyone who tries to utilize the system for a purpose will try to access it using an interface. Indeed, our system also has a user interface built to facilitate users to utilize the services we provide. Users in our system can login/signup using the interface provided to them. They can view all the existing books in our database. The books that are extracted from the datasets are stored in a database. They can search any book by its title or by its author. Users can also view the books they have rated, and they can also log out themselves. Web application interface is what we call as the front-end of our project. This can be accessed from any browser. The interface has been built using Django Framework.

CHAPTER-4 EXPERIMENTAL ANALYSIS AND RESULTS

4.1 System Requirements

A requirement is a feature that the system must have or a constraint that it must to be accepted by the client. Requirement Engineering aims at defining the requirements of the system under construction. Requirement Engineering include two main activities requirement elicitation which results in the specification of the system that the client understands and analysis which in analysis model that the developer can unambiguously interpret. A requirement is a statement about what the proposed system will do. Requirements can be divided into two major categories:

- Functional Requirements.
- Non-Functional Requirements.

4.1.1 Functional Requirements:

A Functional Requirement is a description of the service that the software must offer. It describes a software system or its component. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Functional Requirements describe the interactions between the system and its environment independent of its application.

- Applying the algorithms on the train data
- Display the recommendations by the model.

4.1.2 Non-Functional Requirements:

Non-Functional Requirements specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

Example of nonfunctional requirement, “how fast does the website load?” Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across

the various agile backlogs.

- Accuracy
- Reliability
- Flexibility

.

4.2 System Configuration

4.2.1 Software Requirements

❖ Software:

- Python Version 3.0 or above
- Django Framework
- MySQL python connector version 8.0 or above

❖ Operating System: Windows 10

❖ Database server: MySQL

❖ Tools: Microsoft Visual Studio, Xampp, Web Browser (Google Chrome or Firefox)

❖ Python Libraries: Numpy, pandas, sklearn, Pickle, Matplotlib, Seaborn

4.2.1.1 Introduction to Python

Python is an interpreted, high-level, general-purpose programming language. Python is simple and easy to read syntax emphasizes readability and therefore reduces system maintenance costs. Python supports modules and packages, which promote system layout and code reuse. It saves space but it takes slightly higher time when its code is compiled. Indentation needs to be taken care while coding.

❖ Python does the following:

- Python can be used on a server to create web applications.

- It can connect to database systems.
- It can also read and modify files.
- It can be used to handle big data and perform complex mathematics.
- It can be used for production-ready software development.

Python has many inbuilt library functions that can be used easily for working with machine learning algorithms. All the necessary python libraries must be pre-installed using “pip” command.

4.2.1.2 Introduction to Django Framework

Django is a Python-based free and open-source web application framework that follows the model–template–views architectural pattern. Django was developed to ease the creation of database driven websites and user reusability of components. It is an advanced Python web framework which allows faster development of secure and maintainable websites.

Django takes great care of the web development, so you can focus on writing your app without having to update the wheel. The core framework of Django is light weight, stand-alone web server for development and testing. The main design principles of Django are DRY– Don’t Repeat Yourself and CRUD – Create Read Update and Delete. It’s designed to feel comfortable and easy- to-learn to those used to working with HTML, like designers and front-end developers. Django provides a powerful form library that handles rendering forms as HTML, validating user- submitted data, and converting that data to native Python types.

4.2.1.3 Python Libraries

NumPy: NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object

- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.

Pandas:

Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is fast and it has high-performance & productivity for users. It provides high-performance and is easy-to-use data structures and data analysis tools for the Python language. Pandas is used in a wide range of fields including academic and commercial domains including economics, Statistics, analytics, etc.

SKLearn:

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It is an open-source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface. Sklearn provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Pickle:

Python pickle module is used for serializing and de-serializing a Python object structure. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script. Pickling is useful for applications where you need some degree of persistency in your data. Your program's state data can be saved to disk, so you can continue working on it later.

Matplotlib:

It is a very powerful plotting library useful for those working with Python and NumPy. And for making statistical interference, it becomes very necessary to visualize our data and Matplotlib is the tool that can be very helpful for this purpose. It provides MATLAB like interface only difference is that it uses Python and is open source.

Seaborn:

Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data. It offers the following functionalities:

- Dataset oriented API to determine the relationship between variables.
- Automatic estimation and plotting of linear regression plots.
- It supports high-level abstractions for multi-plot grids.
- Visualizing univariate and bivariate distribution.

4.2.2 Hardware Requirements

1. RAM: 4 GB or above
2. Storage: 30 to 50 GB
3. Processor: Any Processor above 500MHz

4.3 Feasibility Study

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

4.3.1 Economic Feasibility:

As system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economic feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies java1.6 open source, there is nominal expenditure and economic feasibility for certain.

4.3.2 Technical Feasibility:

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team can convert the ideas into working systems. Technical feasibility also involves evaluation of the hardware, software, and other technology requirements of the proposed system. This assessment is based on an outline design of system requirements, to determine whether the company has the technical expertise to handle completion of the project. When writing a feasibility report, the following should be taken to consideration:

- A brief description of the business to assess more possible factors which could affect the study
- The part of the business being examined
- The human and economic factor
- The possible solutions to the problem at this level, the concern is whether the proposal is both technically and legally feasible (assuming moderate cost). The technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. It is an evaluation of the hardware and software and how it meets the need of the proposed system.

4.3.3 Operational Feasibility:

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented
- Will there be any resistance from the user that will undermine the possible application benefits? This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

4.4 Sample Code

a)collaborative

```
filtering.py import numpy
```

```
as np import pandas as pd
```

```
from google.colab import drive drive.mount('/content/drive',force_remount = True) ratings =
```

```
pd.read_csv('/content/drive/MyDrive/Datasets/ratings.csv', sep=',', error_bad_lines=False,
```

```
encoding="latin-1") 34 books =
```

```
pd.read_csv('/content/drive/MyDrive/Datasets/book_data2.csv', sep=',', error_bad_lines=False,
```

```
encoding="latin-1") display(ratings.head()) display(books.head()) #ratings =
```

```
ratings.iloc[:400000,:] ratings.shape # Merge the two tables then pivot so we have Users X
```

```
Books dataframe. ratings_title = pd.merge(ratings, books[['book_id', 'book_title']],
```

```

on='book_id' ) user_book_ratings = pd.pivot_table(ratings_title, index='user_id', columns=
'book_title', values='rating') print('dataset dimensions: ', user_book_ratings.shape, '\n\nSubset
example:') user_book_ratings.iloc[:25, :10] # Drop users that have given fewer than 100 ratings
of these most-rated books user_book_ratings = user_book_ratings.dropna(thresh=100)
print('dataset dimensions: ', user_book_ratings.shape, '\n\nSubset example:')
user_book_ratings.iloc[:25, :10] from sklearn.decomposition import TruncatedSVD # replace
NaN's with zeroes for Truncated SVD user_book_ratings_without_nan =
user_book_ratings.fillna(0) 35 tsvd = TruncatedSVD(n_components=200, random_state=42)
user_book_ratings_tsvd =
tsvd.fit(user_book_ratings_without_nan).transform(user_book_ratings_without_nan)
print('Original number of features:', user_book_ratings_without_nan.shape[1]) print('Reduced
number of features:', user_book_ratings_tsvd.shape[1]) print('Explained variance ratio:',
tsvd.explained_variance_ratio_[0:200].sum()) # view result in a Pandas dataframe, applying the
original indices indices = user_book_ratings.index book_ratings_for_clustering =
pd.DataFrame(data=user_book_ratings_tsvd).set_index(indices) print('dataset dimensions: ',
book_ratings_for_clustering.shape, '\n\nSubset example:')
book_ratings_for_clustering.iloc[:25, :10] from sklearn.model_selection import train_test_split
book_ratings_training, book_ratings_testing = train_test_split(book_ratings_for_clustering,
test_size=0.20, random_state=42) print('Training data shape: ', book_ratings_training.shape)
print('Testing data shape: ', book_ratings_testing.shape) book_ratings_testing.head() 36 # find
the per-book ratings of the test set indices = book_ratings_testing.index test_set_ratings =
user_book_ratings.loc[indices] test_set_ratings.head() mean_ratings_for_random_10 = [] # for
each user, pick 10 books at random that the reader has rated and get the reader's average score

```



```

for those books for index, row in test_set_ratings.iterrows(): ratings_without_nas =
row.dropna() random_10 = ratings_without_nas.sample(n=10) random_10_mean =
random_10.mean() mean_ratings_for_random_10.append(random_10_mean) # get the mean of
the users' mean ratings for 10 random books each mean_benchmark_rating =
sum(mean_ratings_for_random_10) / len(mean_ratings_for_random_10) print('Mean rating for
10 random books per test user: ', mean_benchmark_rating) # trying with the training data after
preprocessing from sklearn.cluster import KMeans clusterer_KMeans =
KMeans(n_clusters=7).fit(book_ratings_training) 37 preds_KMeans =
clusterer_KMeans.predict(book_ratings_training) from sklearn.metrics import silhouette_score
kmeans_score = silhouette_score(book_ratings_training, preds_KMeans) print(kmeans_score) #
trying with the training data after preprocessing from sklearn.mixture import GaussianMixture
clusterer_GMM = GaussianMixture(n_components=7).fit(book_ratings_training) preds_GMM
= clusterer_GMM.predict(book_ratings_training) GMM_score =
silhouette_score(book_ratings_training, preds_GMM) print(GMM_score) indices =
book_ratings_training.index preds = pd.DataFrame(data=preds_KMeans,
columns=['cluster']).set_index(indices) preds.head() # get a list of the highest-rated books for
each cluster def get_cluster_favorites(cluster_number): # create a list of cluster members
cluster_membership = preds.index[preds['cluster'] == cluster_number].tolist() # build a
dataframe of that cluster's book ratings 38 cluster_ratings =
user_book_ratings.loc[cluster_membership] # drop books that have fewer than 10 ratings by
cluster members cluster_ratings = cluster_ratings.dropna(axis='columns', thresh=10) # find the

```

```

cluster's mean rating overall and for each book means = cluster_ratings.mean(axis=0) # sort
books by mean rating favorites = means.sort_values(ascending=False) return favorites # for
each cluster, determine the overall mean rating cluster members have given books def
get_cluster_mean(cluster_number): # create a list of cluster members cluster_membership =
preds.index[preds['cluster'] == cluster_number].tolist() # create a version of the original ratings
dataset that only includes cluster members cluster_ratings =
ratings[ratings['user_id'].isin(cluster_membership)] # get the mean rating return
cluster_ratings['rating'].mean() cluster0_books_storted = get_cluster_favorites(0) cluster0_mean
= get_cluster_mean(0) print('The cluster 0 mean is:', cluster0_mean)
cluster0_books_storted[0:10] 39 cluster1_books_storted = get_cluster_favorites(1)
cluster1_mean = get_cluster_mean(1) print('The cluster 1 mean is:', cluster1_mean)
cluster1_books_storted[0:10] cluster2_books_storted = get_cluster_favorites(2) cluster2_mean
= get_cluster_mean(2) print('The cluster 2 mean is:', cluster2_mean)
cluster2_books_storted[0:10] cluster3_books_storted = get_cluster_favorites(3) cluster3_mean
= get_cluster_mean(3) print('The cluster 3 mean is:', cluster3_mean)
cluster3_books_storted[0:10] cluster4_books_storted = get_cluster_favorites(4) cluster4_mean
= get_cluster_mean(4) print('The cluster 4 mean is:', cluster4_mean)
cluster4_books_storted[0:10] 40 cluster5_books_storted = get_cluster_favorites(5)
cluster5_mean = get_cluster_mean(5) print('The cluster 5 mean is:', cluster5_mean)
cluster5_books_storted[0:10] cluster6_books_storted = get_cluster_favorites(6) cluster6_mean
= get_cluster_mean(6) print('The cluster 6 mean is:', cluster6_mean)

```

```

cluster6_books_storted[0:10] # associate each test user with a cluster test_set_preds =
clusterer_KMeans.predict(book_ratings_testing) test_set_indices = book_ratings_testing.index
test_set_clusters = pd.DataFrame(data=test_set_preds,
columns=['cluster']).set_index(test_set_indices) test_set_clusters
mean_ratings_for_cluster_favorites = [] # put each cluster's sorted book list in an array to
reference 41 cluster_favorites = [cluster0_books_storted, cluster1_books_storted,
cluster2_books_storted, cluster3_books_storted, cluster4_books_storted,
cluster5_books_storted, cluster6_books_storted] # for each user, find the 10 books the reader
has rated that are the top-rated books of the cluster. # get the reader's average score for those
books for index, row in test_set_ratings.iterrows(): user_cluster = test_set_clusters.loc[index,
'cluster'] favorites = cluster_favorites[user_cluster].index user_ratings_of_favorites = [] #
proceed in order down the cluster's list of favorite books for book in favorites: # if the user has
given the book a rating, save the rating to a list if np.isnan(row[book]) == False:
user_ratings_of_favorites.append(row[book]) # stop when there are 10 ratings for the user if
len(user_ratings_of_favorites) >= 10: break # get the mean for the user's rating of the cluster's
10 favorite books mean_rating_for_favorites = sum(user_ratings_of_favorites) /
len(user_ratings_of_favorites)
mean_ratings_for_cluster_favorites.append(mean_rating_for_favorites) mean_favorites_rating
= sum(mean_ratings_for_cluster_favorites) / len(mean_ratings_for_cluster_favorites) 42
print('Mean rating for 10 random books per test user: ', mean_benchmark_rating) print('Mean
rating for 10 books that are the cluster\'s favorites: ', mean_favorites_rating) print('Difference

```

```

between ratings: ', mean_favorites_rating-mean_benchmark_rating) from sklearn.metrics import
mean_squared_error rmse =
mean_squared_error(mean_ratings_for_random_10,mean_ratings_for_cluster_favorit
es,squared = False) # taking root of mean squared error print(rmse) accuracy = 1.96 * rmse
print(accuracy) import random def recommend(cluster_assignments, user_id): user_cluster =
cluster_assignments favorites = get_cluster_favorites(user_cluster).index favorites =
random.choices(favorites, k=10) return favorites recommendation8667 = recommend(5, 8667)
print(recommendation8667) b) contentbasedrecommender.py from google.colab import drive
43 drive.mount('/content/drive',force_remount = True) # importing libraries import pandas as pd
from sklearn.metrics.pairwise import linear_kernel from sklearn.feature_extraction.text import
TfidfVectorizer # reading file book_description =
pd.read_csv('/content/drive/MyDrive/Datasets/book_data2.csv', sep=',', error_bad_lines=False,
encoding="latin-1") # checking if we have the right data book_description.head() # removing
the stop words books_tfidf = TfidfVectorizer(stop_words='english') # replace NaN with empty
strings book_description['book_desc'] = book_description['book_desc'].fillna("") # computing
TF-IDF matrix required for calculating cosine similarity book_description_matrix =
books_tfidf.fit_transform(book_description['book_desc']) # Let's check the shape of computed
matrix book_description_matrix.shape # compuing cosine similarity matrix using linear_kernel
of sklearn cosine_similarity = linear_kernel(book_description_matrix,
book_description_matrix) 44 # Get the pairwsie similarity scores of all books compared to the
book passed by index, sorting them and getting top 5 similarity_scores =

```

```
list(enumerate(cosine_similarity[1])) similarity_scores = sorted(similarity_scores,  
key=lambda x: x[1], reverse=True) similarity_scores = similarity_scores[1:6] # Get the similar  
books index books_index = [i[0] for i in similarity_scores] # printing the top 5 most similar  
books using integer-location based indexing (iloc) print  
(book_description['book_title'].iloc[books_index])
```

4.5 Experimental analysis and Performance Measures

The books dataset which initially has 2080 features in it has been reduced to 200 features after using Truncated SVD in feature extraction. This reduced dataset has been used to build the models. By calculating the coefficient of silhouette, the quality of clustering of different trained models can be compared. A sample of the training dataset is shown in below fig-4.1.

After building the model using the training data for clustering, the next step is to measure the performance of the model. To evaluate the efficacy of the model, silhouette score is calculated.

4.5.1. Performance Analysis and Models Comparison

Out of all the trained models we need to choose the best model. We need to analyze the performance of each model and then compare the silhouette scores of the two trained models.



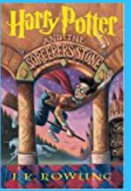

The above Figure 4.3 shows a screenshot of notebook with silhouette score for K-Means clustering model. This coefficient of silhouette of this model is found out to be 0.02688953262460168, which is positive that means the clustering is good and appropriate.

5. RESULTS (Screenshots)

To demonstrate the results of our project, we have determined the cluster favorites to recommend the book to the user of a particular cluster. This recommendation is carried out in a python notebook.

[My Book recommender](#) [Home](#) [Recommend](#) [Contact](#)

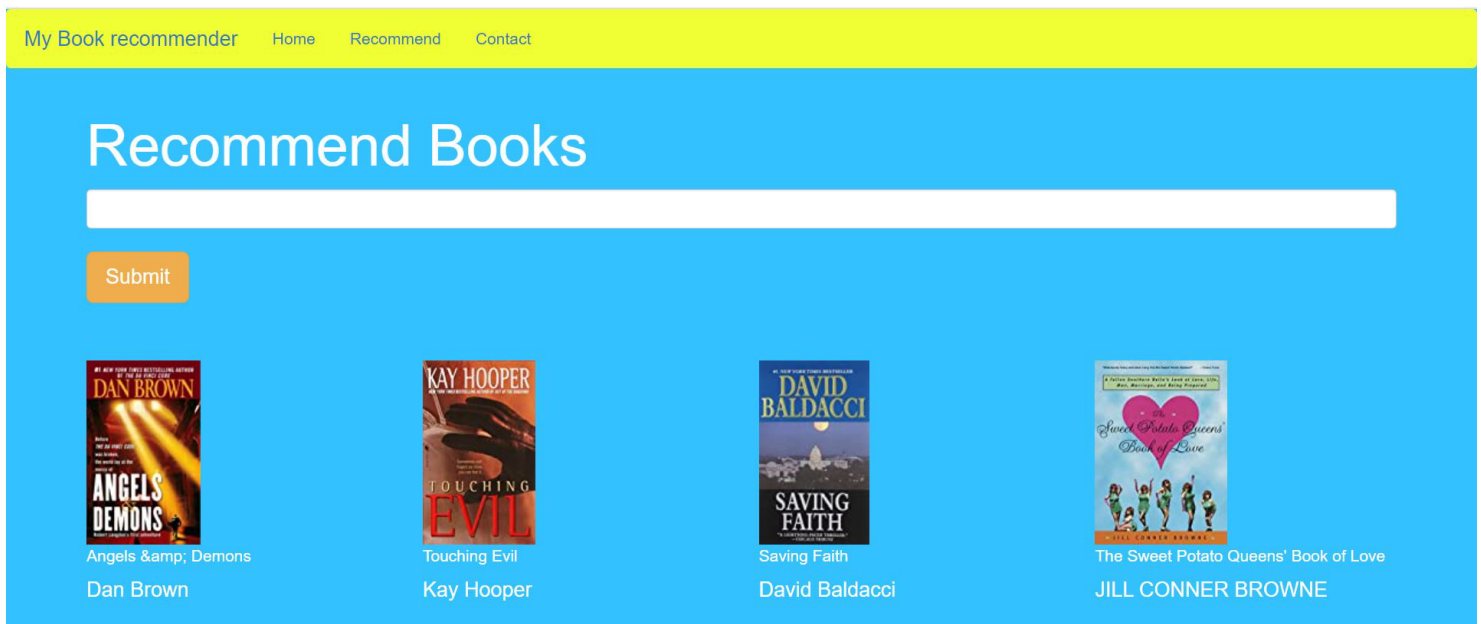
Top 50 Books

			
Harry Potter and the Prisoner of Azkaban (Book 3)	Harry Potter and the Goblet of Fire (Book 4)	Harry Potter and the Sorcerer's Stone (Book 1)	Harry Potter and the Order of the Phoenix (Book 5)
J. K. Rowling	J. K. Rowling	J. K. Rowling	J. K. Rowling
Votes - 428	Votes - 387	Votes - 278	Votes - 347
Rating - 5.852803738317757	Rating - 5.8242894056847545	Rating - 5.737410071942446	Rating - 5.501440922190202

[My Book recommender](#) [Home](#) [Recommend](#) [Contact](#)

Recommend Books

Submit



Now, we made recommendations for a user as shown in the fig 4.8. As like the above data, every time we want recommendations, we need to continue the whole process. But the users who are not familiar with programming or python notebooks will find it difficult to do the whole thing. To get rid of this type of problems we need to have a user interface from which everyone can access and use our system at ease. For our system, we have built User interface i.e., our front-end using Django web framework. We have created some Html pages and connected them to our python modules using Django. Users have login credentials which are used to login into website. To see how our interface works, we have taken some screenshots while checking the functionalities our system.

The following screenshots will demonstrate the final results of our project:

- A new user can be signed up through this page. In Figure 4.9, we can have a clear look at this page. There is button with the name register at the bottom of the page. After filling all the details which are required to register, click on the register button. Then it will be redirected to a python module to which it is interfaced
- Using the page shown in Figure 4.9, we have registered a new user. In the figure 4.10 we can see the login page in which user can login into Home page with login credentials. If we didn't register, we cannot login to the website.
- The recommendation of books for users is shown in this page i.e. Home page. The

recommendations shown in this figure based on collaborative filtering system. This is the page that we have designed where our users can search for any book which available in our database. The page can be seen in the figure 4.11. This page can be accessed with user login.

- Under the Recommended books, there are popular books which are recommended by the Content Based Filtering system are also shown in the home page, which are named as popular books. These books were shown to the new users also irrespective of the user data. We can see the books in Figure 4.12.
- After signing into our site, users can search for any book by entering the details of that particular book. Users can search book by using book author or book title. We can see the details of the book after searching with book title in fig 4.13.
- Just like by searching the book with book title, user can also search the book by its author name. We can see the details of the book after searching with book author name in fig 4.14.
- We can give rating to any book whichever we want after clicking on the book. After clicking on the book, we are redirected to another page where we can give the rating to the book. Figure 4.15 shows 5 black stars under the details of the book where we can give rating by clicking on the stars. After giving rating and clicking on the submit button, our rating to the book is stored.
- After giving rating to the book, we can see that book in another page which is redirected after clicking on the Show rated books button. There we can see the books which are rated by us in sequential order. Figure 4.16 shows the screenshot of the books which are rated by user.

CHAPTER-6 CONCLUSION

Conclusion

In this project, we have recommended the books for a user using the model trained using K-Means Clustering which is a Collaborative Filtering Technique. We have also compared different models built using different methods and identified the best model and justifies why it has chosen that model. We have used the books dataset that is available in the Goodreads website which consists of more than 3000 books. The models are built using the reduced features which is done by Truncated SVD. Based on those features the author built a model that gives a positive Silhouette score. The model that is suggested by this paper is useful for book readers. The system we have developed can make recommendations for new users also.

CHAPTER-7 FUTURE ENHANCEMENT

The System has adequate scope for modification in future if it is necessary. Development and launching of Mobile app and refining existing services and adding more service, System security, data security and reliability are the main features which can be done in future. The API for the shopping and payment gateway can be added so that we can also buy a book now. In the existing system there are only some selected categories, so as an extension to the site we can add more categories as compared to existing site. Also, we can add admin side with some functionalities like books management, User management etc.

CHAPTER - 8 REFERENCES

1. [Book Recommendation System | Aman Kharwal \(thecleverprogrammer.com\)](#)
2. [Book Recommendation System | Build A Book Recommendation System \(analyticsvidhya.com\)](#)
3. [My Journey to building Book Recommendation System. . . | by Chhavi Saluja | Towards Data Science](#)
4. [Building A Book Recommender System – The Basics, kNN and Matrix Factorization | DataScience+\(datascienceplus.com\)](#)

