

A FIELD PROJECT REPORT

on

**“Intelligent Book Recommendation System Using Machine Learning Techniques”**

**Submitted**

by

221FA04041

P. Bhagya Sri

221FA04570

K. Jaya Sri

221FA04149

G. Sindhu Sri

221FA04588

V. Pujitha

**Under the guidance of**

*Ms. Sajida Sultana. Sk*

*Assistant Professor*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**  
**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH**  
**Deemed to be UNIVERSITY**  
**Vadlamudi, Guntur.**  
**ANDHRA PRADESH, INDIA, PIN-522213.**

### **CERTIFICATE**

This is to certify that the Field Project entitled “**Intelligent Book Recommendation System Using Machine Learning Techniques**” that is being submitted by 221FA04041 (P. Bhagya Sri), 221FA04149 (G. Sindhu Sri), 221FA04570 (K. Jaya Sri), 221FA04588 (V. Pujitha) for partial fulfilment of Field Project is a bonafide work carried out under the supervision of Ms. Sajida Sulthana.Sk M.Tech., Assistant Professor, Department of CSE.

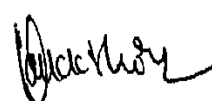
Guide name& Signature



Dr. S. V. Phani Kumar

Assistant/Associate/Professor,  
CSE

HOD,CSE



Dr.K.V. Krishna Kishore

Dean, SoCI



## DECLARATION

We hereby declare that the Field Project entitled **“Intelligent Book Recommendation System Using Machine Learning Techniques”** is being submitted by 221FA04041 (P. Bhagya Sri), 221FA04149 (G. Sindhu Sri), 221FA04570 (K. Jaya Sri), 221FA04588 (V. Pujitha) in partial fulfilment of Field Project course work. This is our original work, and this project has not formed the basis for the award of any degree. We have worked under the supervision of Ms. Sajida Sulthana Sk., M.Tech., Assistant Professor, Department of CSE.

By

**221FA04041 (P. Bhagya Sri),**

**221FA04149 (G. Sindhu Sri),**

**221FA04570 (K. Jaya Sri),**

**221FA04588 (V. Pujitha)**

Date:

# ABSTRACT

In this paper, a book recommendation system, generically advised using multiple machine learning models like Random Forest, Naive Bayes, Decision Tree, and Ridge Classifiers to analyze user preferences to make predictions about book recommendations is developed. A hybrid model was built in order to overcome the limitation of each and every algorithm so that better accuracy and relevance were found out in recommendations. This hybrid model integrates Singular Value Decomposition (SVD) with K-Nearest Neighbours (KNN) by integrating collaborative filtering to address user-item interaction along with similarity-based recommendations that suggest books similar in user preference. The model, therefore, showed relevance, with a Precision score of 89.35%, Recall of 59.01%, and F1 Score of 71.30%, serving as standard references that balance the recommendations in terms of accuracy as well as relevance and efficiency. In addition, finally, this hybrid approach will provide a more reliable solution to improve user satisfaction in book recommendation systems by remedying the deficiency of single-algorithm models and further improving the quality of its service.

**Keywords:** Book Recommendation System ,Random Forest, Naive Bayes, Decision Tree, Ridge Classifier, Hybrid Model, Singular Value Decomposition (SVD).K-Nearest Neighbors (KNN).

## TABLE OF CONTENTS

1. Introduction	1
1.1 What is Book Recommendation System?	2
1.2 Importance of Book Recommendation System	2
1.3 Role of Machine Learning in Book Recommendation System	2
1.4 Challenges in building Recommendation Systems	3
1.5 Applications of book recommendation systems	3
2. Literature Survey	4
2.1 Literature review	9
2.2 Motivation	9
3. Proposed System	10
3.1 Input dataset	12
3.1.1 Detailed features of dataset	13
3.2 Data Pre-processing	15
3.2.1 Random Sampling	16
3.2.2 Data Merging	16
3.2.3 Handling Missing Values	16
3.2.4 Feature Selection	16
3.3 Model Building	16
3.3.1 Hybrid of SVD and KNN	
3.3.2 Decision Tree	
3.3.3 Naïve Bayes	
3.3.4 Ridge Classifier	
3.3.5 Random Forest	
3.4 Methodology of the system	18
3.5 Model Evaluation	21
3.6 Performance Metrics	21
4. Implementation	24
4.1 Environment Setup	25
4.2 Sample code for preprocessing and Hybrid Model	25
5. Experimentation and Result Analysis	27
6. Conclusion	36
7. References	38

## LIST OF FIGURES

Figure 3.1 Books Dataset	12
Figure 3.2 Ratings Dataset	12
Figure 3.3 Booktags Dataset	12
Figure 3.4 Tags Dataset	12
Figure 3.5 Read Dataset	12
Figure 3.6 Flow Diagram for Data preprocessing	15
Figure 3.7 Flow Diagram for Hybrid SVD and KNN Model	18
Figure 4.1 Code for Data Preprocessing	26
Figure 4.2 Code for Hybrid model (SVD+KNN)	26
Figure 5.1 Top Recommendations	28
Figure 5.2 Top Recommendations by Hybrid Model(SVD+KNN)	29
Figure 5.3 Top Recommended books with Combined scores	29
Figure 5.4 User Feedback Summary	30
Figure 5.5 Performance of Hybrid Model	30
Figure 5.6 Confusion Matrix	31
Figure 5.7 ROC Curve	31
Figure 5.8 Performance of Decision Tree	32
Figure 5.9 Confusion Matrix (Decision Tree)	32
Figure 5.10 ROC Curve (Decision Tree)	33
Figure 5.11 Comparison of all Models	35

## **LIST OF TABLES**

Table 3.1 Performance Comparison of Recommendation System	23
---	----

# **CHAPTER-1**

## **INTRODUCTION**



# **1. INTRODUCTION**

## **1.1 What is Book Recommendation System?**

A book recommendation system is a tool that recommends books based on a user's preferences, reading history, and behaviors. It uses collaborative filtering and content-based filtering to improve the reading experience. Hybrid approaches enhance the accuracy of recommendations. Despite challenges like cold start with new users and diverse recommendations, book recommendation systems are widely used in online retailers and libraries.

## **1.2 Importance of Book Recommendation System**

The importance of book recommendation systems lies in their ability to provide personalized suggestions that enhance user experience and engagement. By analyzing individual preferences and reading histories, these systems help users discover new books that align with their interests. As a result, they improve user retention for digital platforms and bookstores, driving sales by efficiently connecting readers to relevant content. Overall, book recommendation systems play a crucial role in fostering a more satisfying and efficient reading journey while benefiting retailers.

## **1.3 Role of Machine Learning in Book Recommendation System**

Machine learning is important in a book recommendation system because it performs an analysis of user data, for example, ratings, reviews, and reading history, plus book metadata like genre and authorship, thus enabling the recommendation of personalization techniques: collaborative filtering and content-based filtering, suggesting titles based on user preferences. Predictive modeling takes such user interest into account, while the matrix factors permit trap learning through sparse user-item interactions. Further supported are hybrid approaches using combined algorithms. It assists in user segmentation based on reading patterns and natural language processing practices, implementing book content and review analysis. This flexible recommendation system results in extensive personal engagement and customization of reading experiences.

## **1.4 Challenges in building Recommendation Systems**

Three problems related to data quality include inconsistencies, missing information, and cold-start; rises in algorithm installation and scalability appear with larger datasets as well. Adjustment to

user preferences and good scalability in size with massive datasets become must-haves for any such system. The most complex step would be to choose the right algorithm and fine-tune its parameters. An effective metric for evaluating results and getting feedback from users is also indispensable. We have to make sure we balance the plug-pull habits with popularity bias and the interpretability of recommendations to gain user trust and engagement. These issues must be properly addressed for the functioning of a robust and user-friendly recommendation system.

### **1.5 Applications of book recommendation systems**

Book recommendation systems play a significant role in various domains by enhancing user engagement and streamlining the discovery process.

- Online Retail
- Public Libraries
- Educational Institutions
- Digital Learning Platforms
- E-book Services

# **CHAPTER-2**

## **LITERATURE SURVEY**

## 2. LITERATURE SURVEY

**Wayesa, Mesfin Leranso, et al[1]** proposed a pattern-based hybrid book recommendation system integrating Content-based Filtering (CBF) with Collaborative Filtering (CF). A hybrid clustering approach assists in grouping books and users into semantically similar categories and applying semantic pattern clustering to improve similarity accuracy. The IR evaluation metrics, among them Precision, Recall, and F1-score, play a crucial role in further enhancing the recommendation quality. The experimental analysis on the GoodBooks dataset reports that the hybrid model has significantly outperformed current approaches with respect to predictive accuracy. Future work will focus on semantic clustering methods to improve performance and scalability and on generalization across other recommender system domains.

In this paper, **Alaa Yaseen Taqa and Abdullah Mohammed Saleh[2]**, present a user-based collaborative recommendation system aimed at enhancing the accuracy of book suggestions by addressing personalization and data sparsity. The model consists of three central modules: data preparation, similarity computation, and prediction using a weighted K-nearest neighbor algorithm. The effective model detects user preferences towards recommendations in an efficient manner using cosine and Pearson-Baseline similarity, where a user-to-user similarity matrix has been identified for better accuracy of recommendations. On-course experiments are conducted on the Book-Crossing dataset, which lowers error rates and shows significant improvement on RMSE and MAE metrics. Future developments are aimed further towards improving the similarity computations and scaling it up to broader digital content platforms, enabling more advanced functionalities for explicit user customization in a wider population of users.

A book recommendation system by **Devika P., R.C. Jisha et al[3]**., based on FPIntersect: Devika, Jisha, and Sajeev introduced a recommendation system that deals with performance issues in book suggestions using the FPIntersect method. Frequent traditional approaches like the Apriori algorithm may have to undertake multiple scans of the database, thus lowering their efficiency. Unlike the others, FPIntersect performs itemset generation on the basis of finding the intersections of user IDs, thus reducing the number of scans and providing a great leap in the speed and accuracy of the system. By this way, and by assigning a score to every book on the basis of user opinions and ratings, importing opinion mining to study user reviews and obtain sentiment polarity became very useful. Based on good ratings, more individualized recommendations of books are made by the transactional database. Simulations show the superiority of FPIntersect to Apriori, especially

in reducing runtime. Future research attempts to improve.

The Opinion Mining Based Book Ranking System of **Rashid Ali, Jamshed Siddiqui et al [4]**, shall therefore design a book recommendation system such that it uses opinion mining to rank books; hence particularly designed for computer science literature. Reviews from users are collected by the system, which assesses them for their essential features: relevance, helpfulness, and material quality. There are weighted ratings of such attributes that this model will generate to create a ranking. There are also the factors of positive and negative reviews in determining the relevant score, depending upon certain features' importance. For instance, availability and relevance are given certain weights for user opinions to affect rating scores. Objections against some aspects (such as price or irrelevant information) are taken off the score. The ranking mechanism presents books efficiently rated by the readers along with very few useful suggestions.

**Sarma, Mittra and Hossain [5]** have developed a Customized Book Suggestion System, based on Cosine Similarity and Clustering, aimed at increasing recommendation accuracy. Contrary to existing systems that rely solely on user-based ratings, this model groups books according to user ratings and preferences. Within each cluster, recommendations for each book are provided to users based on similar interests. Clustering successfully addresses shortcomings of collaborative filtering and yields more accurate recommendations. Model validation with metrics such as: sensitivity, specificity and F-score; Specificity indicates the system's effectiveness in filtering out irrelevant recommendations. The model was further validated by a detailed examination using ROC curves whereby emphasis was added on clustering as an effective strategy capable of generating recommendations tailored to users' individual needs.

**Devika PV, Jyothisree K, et al [6]**, Using Collaborative Filtering (CF) enhanced by Pearson Correlation and K-Nearest Neighbours (KNN), the authors propose the present method to enhance the accuracy of recommendations. The method recommends books matching both fresh and old users' interests and preferences based on Book-Book similarity measures for the purposes of recommending books of interest. It addresses commonly occurring problems regarding sparsity and "cold start" for new users by assigning priority to the highly-rated books based on ratings in huge datasets. In turn, it generates personalized and relevant recommendations pertaining to book choices.

**Praveena Mathew, Bincy Kuriakose et al [7]**, advise a hybrid recommendation strategy that unanimously combines content-based filtering. Collaborative filtering, and association rule

mining. The technology has the functionality of keyword recommendation to find books that meet user wants by assessing item attributes and browsing behaviors. By combining these approaches, it is assured that recommended books represent individual patterns, thus improving recommendation quality, user engagement, and satisfaction. Their further scalable version intends to show improved performance across a wide range of recommendation settings.

**Khalid Anwar, Jamshed Siddiqui, et al [8]**, focus on supervised and unsupervised models like SVM and KNN. They found that based on rating and preference patterns, they found sentiment analysis and clustering to look into trends that can better recommend books. Their approach would be to increase accuracy for recommendation, which could let e-commerce and library book selectors maximize stock while creating an enhanced user experience. In the near future, ML capabilities would be built to handle increasingly complex data sets with aims to tailor recommendations even better to multiple user populations.

**Anwar and Uma [9]** introduced CD-SPM, which is a book recommendation system. It is a cross-domain recommendation system that is an integration of sequential pattern mining, collaborative filtering, and semantic similarity to evade the data sparsity problem and the cold-start problem. In this system, Wpath and Ontology are used to quantify the semantic similarity for different categories like novels and movies. With this, it enhances the mapping between domains through the evaluation of similarity while the Topseq rules generate common sequences for accurate item prediction. Experimental performance evaluations demonstrated the superiority of CD-SPM over traditional techniques of CF. Future works will be directed toward improving algorithms for semantic clustering and applications of the system in broad domains.

**Tsujia et al[10]**, proposed a book recommendation system combining bibliographic information and machine learning algorithms using library lending records. Some of the features integrated in the SVM model of the recommendation system include title similarity, association rules, and categorization matches. In order to increase the accuracy of suggestions, according to their study, their research should enhance the happiness of the customer because they are providing more individualized and relevant book recommendations compared to Amazon's recommendation system. In order to increase the accuracy of the suggestion, according to the study, the future research can focus on the optimization of the feature weighting method.

**Seongwoo Cho, Jongsu Park et al[11]**, proposed a Fine-Tuned Retrieval Augmented Language

Model for machine tool operation support. Technical queries improved from 51% to 84% accuracy with the help of the system that used fast engineering, retrieval augmented generation, and fine-tuning. The model applied on a local server demonstrates tremendous improvement in predictive and qualitative accuracy and offers real-time contextual support to less experienced operators.

**Liu Ye, Yin Yimeng et al [12]**, online reviews in Douban Books to observe the perception of teaching books. They study the perception aspect of content quality-a highly critical factor affecting users' impression-by using aspect level sentiment analysis and topic analysis that can be implemented with Python. They used LDA on topic clustering, sentiment triplet extraction, and keyword extraction. This attempt at sentiment analysis tries to present it to publishers as a starting point for enhancing the quality of books and publishing standards in general.

**Sushama Rajpurkar, Darshana Bhatt et al [13]**, proposed a hybrid book recommendation system that combined CBF, CF, and ARM. This solution addressed several common challenges in the traditional methods, like sparsity of user preference and lack of assessment about content quality. Two steps of the workflow of the system are to identify categories of books through user purchasing history and make recommendations with the help of CBF for books falling into the same category. Association Rule Mining finds frequent books in each category, after the CBF approach, while item-based CF improves recommendation by checking rating from a similar customer. Experimental results indicated that the recommended approach not only increases the user's satisfaction but also the precision of recommendations while storing it offline for efficient purposes. Real-time data update along with other recommendation parameters, like diversity and novelty, can be a scope for future study.

**Anant Duhan and N. Arunachalam [14]**, develop a machine learning-based book recommendation system based on K-Nearest Neighbour classifiers, Decision Trees, and Logistic Regression. Their strategy involves both collaborative filtering and content-based filtering for the typical problems of data sparsity and the cold-start problem. F1-score, precision, and recall in evaluating the model will prove essential in validating the potential of this model in its capability to provide precise recommendations. Further work includes an augmentation of data sources with better scalability, adding dimensions like temporal, and incorporating awareness of context.

**Francesco Osborne, Thiviyan Thanapalasingam et al [15]**, and present the Smart Book Recommender (SBR), a semantic recommendation engine. SBR utilizes a precomputed similarity measure to match conference themes with Springer Nature books, journals, and proceedings by

making use of a large-scale ontology of Computer Science topics. For scholarly publications, this approach enhances marketing effectiveness and recommendation accuracy. Further developments include incorporating more features such as sales data and optimizing the user interface for wider academic use.

## **2.1 Motivation**

A book recommendation system is crucial due to the vast and diverse range of books available, making it difficult for readers to find content tailored to their interests. With digitization, users are overwhelmed by choices, leading to decision fatigue and limited exposure to enriching reads. A well-designed recommendation system enhances user engagement and satisfaction by generating personalized suggestions, promoting lesser-known titles, and fostering a deeper connection with literature.



# **CHAPTER-3**

## **PROPOSED SYSTEM**

### **3. PROPOSED SYSTEM**

Book recommendation systems are meant to help users find books compatible with their interests and reading preferences, including some aspects that are not covered by classical popularity methods. The hybrid model integrates both SVD and KNN to optimize the accuracy and relevance of the recommendations. The system begins by collecting user interaction data to create a user-item interaction matrix in which users rate books. SVD will decompose this matrix into latent factors that capture hidden properties of the user and items. This enables the system to predict the ratings of the books for which users have not yet given ratings. KNN then determines the similarity between users or books based on the latent features by calculating distances in the feature space of the users or objects. The K nearest neighbors can then be chosen in order to identify those users with tastes in common or books that are contextually similar. Recommendations combine the SVD-based predictions and KNN insights ranked by their expected ratings. Feedback from users is also part of the system, which enables its improvement and address common problems such as the cold-start problem and data sparsity at the same time as ensuring that the user experience is personalized and engaging.

### 3.1 Input dataset

The dataset used in this project is taken from goodreads from Kaggle. The key Datasets are:

- Books Dataset: This Dataset contains information about Books such as book\_id, goodreads\_book\_id, title, author, isbn, isbn\_code.
- Ratings Dataset: This Dataset contains attributes user\_id, book\_id, rating.
- Book\_tags Dataset: This Dataset contains attributes goodreads\_book\_id, tag\_id, count.
- Tags Dataset: This Dataset contains attributes tag\_id, tag\_name.
- Read Dataset: This Dataset contains the attributes user\_id, book\_id.

TABLE 1

book_id	goodreads_book_id	best_book_id	work_id	books_count	isbn	isbn13
1	2767052	2767052	2792775	272	4.39E+08	9.78E+12
2	3	3	4640799	491	4.4E+08	9.78E+12
3	41865	41865	3212258	226	3.16E+08	9.78E+12

Figure 3.1 Books Dataset

TABLE 2

user_id	book_id	rating
1	258	5
2	4081	4
2	260	5

Figure 3.2 Ratings Dataset

TABLE 3

goodreads_book_id	tag_id	count
1	30574	167697
1	11305	37174

Figure 3.3 Book Tags Dataset

TABLE 4

Tag_id	tag_name
0	-
1	--1-

Figure 3.4 Tags Dataset

TABLE 5

user_id	book_id
9	8
15	398

Figure 3.5 Read Dataset

### 3.1.1 Detailed Features of the Dataset

Features of Books Dataset:

- **book\_id**: A unique identifier for each book.
- **goodreads\_book\_id**: Unique identifier for the book on Goodreads.
- **best\_book\_id**: Alternative unique identifier for the best or primary version of the book.
- **work\_id**: Identifier for the book's work (common across different editions).
- **books\_count**: Number of editions or versions of the book available.
- **isbn**: The International Standard Book Number for the book.
- **isbn13**: The 13-digit ISBN of the book.
- **authors**: Names of authors for the book.
- **original\_publication\_year**: Year the book was originally published.
- **original\_title**: The original title of the book.
- **title**: Current title of the book.
- **language\_code**: Language code representing the book's primary language.
- **average\_rating**: Average rating given by readers.
- **ratings\_count**: Total number of ratings received by the book.
- **work\_ratings\_count**: Total ratings across all editions or versions.
- **work\_text\_reviews\_count**: Total text reviews across all editions.
- **ratings\_1**: Number of 1-star ratings.
- **ratings\_2**: Number of 2-star ratings.
- **ratings\_3**: Number of 3-star ratings.
- **ratings\_4**: Number of 4-star ratings.
- **ratings\_5**: Number of 5-star ratings.
- **image\_url**: URL for the book's cover image.
- **small\_image\_url**: URL for a smaller version of the book's cover image.

Features of Book\_tags Dataset:

- **goodreads\_book\_id**: A unique identifier for each book on Goodreads.
- **tag\_id**: A unique identifier for each tag, linking to specific descriptors or genres for the book.
- **count**: The number of times this tag has been associated with the book.

Features of Tags Dataset:

- **tag\_id**: A unique identifier for each tag, represented as a sequential integer.

- **tag\_name**: The name or label associated with each tag, possibly encoding specific attributes.

Features of Read Dataset:

- **user\_id**: Represents the unique identifier of each user in the dataset.
- **book\_id**: Represents the unique identifier of each book associated with each user.

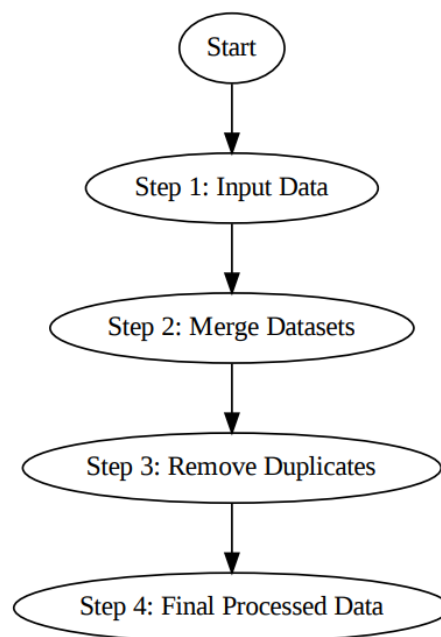
Features of Ratings Dataset:

- **user\_id**: Unique identifier for each user.
- **book\_id**: Unique identifier for each book rated by the user.
- **rating**: Rating given by the user for a specific book, typically on a scale (e.g., 1-5)

### 3.2 Data Pre-processing

Data pre-processing is the essential process of preparing raw data for analysis and modelling by cleaning, transforming, and structuring it to enhance data quality and utility. It involves tasks like handling missing values, correcting errors, encoding features, and scaling data to ensure it's in an optimal form for further analysis. It encompasses a range of operations and transformations designed to refine raw data, ensuring that it is clean, structured, and amenity subsequent analysis. This process is driven by its manifold significance in data science and analysis.

Through meticulous data cleaning, transformation, feature engineering, dimensionality reduction, outlier handling, scaling, and data splitting, it prepares raw data for more accurate and reliable analysis and modelling. Ultimately, the goal is to obtain more meaningful insights, make informed decisions, and optimize predictive models for a wide range of applications in data science and analysis.



**Fig 3.6 Flow Diagram of Data Preprocessing**

#### 3.2.1 Random Sampling:

- To manage computational load and avoid session crashes when working with large datasets, a random sample of 10% was taken from the ratings dataset. This sampling was achieved using the `sample()` method with a fixed `random_state` to ensure reproducibility.

### 3.2.2 Data Merging:

- The procedure involves removing missing values from a merged dataset by dropping null-entered rows, simplifying data handling and retaining existing entries, but it may result in data loss.

### 3.2.3 Handling Missing Values:

- Missing values within the merged dataset were addressed by dropping any rows containing null entries. This simplified data handling by ensuring that only complete data entries were retained, though it may lead to some data loss.

### 3.2.4 Feature Selection:

- It is a data preprocessing technique for selecting a subset of relevant features (or columns) from a dataset that contribute most to the predictive power or quality of a model.

## 3.3 Model Building

### 3.3.1 Hybrid of SVD and KNN

The hybrid model combines Singular Value Decomposition (SVD) for dimensionality reduction with K-Nearest Neighbors (KNN) for making recommendations based on similarities.

**SVD:**

$$R = U\Sigma V^T$$

Where:

- $R$  is the original user-item rating matrix.
- $U$  is the user matrix.
- $\Sigma$  is the diagonal matrix of singular values.
- $V^T$  is the item matrix.

**KNN:**

$$d(q, q') = \sqrt{\sum_{i=1}^n (q_i - q'_i)^2}$$

Where  $d(q, q')$  is the Euclidean distance between points  $q$  and  $q'$ , determining the nearest neighbors.

The combination uses SVD to reduce dimensionality and KNN to find similar books based on these lower-dimensional features.

### 3.3.2 Decision Tree

Decision Trees split the data into branches based on feature values to make a prediction.

$$Gini = 1 - \sum_{i=1}^c (p_i)^2$$

Where:

- Gini is the impurity measure.
- $p_i$  is the probability of class  $i$ .

The model selects the features that maximize the information gain at each split.

### 3.3.3 Naive Bayes

Naive Bayes assumes independence between predictors and uses Bayes' Theorem:

$$P(y/X) = \frac{p(x/y)P(y)}{P(x)}$$

Where:

- $P(y|X)$  is the posterior probability of class  $y$  given feature set  $X$ .
- $P(X|y)$  is the likelihood.
- $P(y)$  is the prior probability of class  $y$ .
- $P(X)$  is the probability of feature set  $X$ .

### 3.3.4 Ridge Classifier

Ridge Classifier is a variant of linear regression for classification with L2 regularization:

$$y = \text{sign}(b + \mathbf{w}^T \mathbf{x})$$

$$\text{Cost} = \frac{1}{2} \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|_2^2$$

Where:

- $b$  is the bias.
- $\mathbf{w}$  is the weight vector.
- $\lambda \|\mathbf{w}\|_2^2$  is the regularization term to avoid overfitting.

### 3.3.5 Random Forest

Random Forest is an ensemble method that combines multiple decision trees:

$$F(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^M f_m(\mathbf{x})$$

Where:

- $F(\mathbf{x})$  is the final prediction.
- $M$  is the number of trees in the forest.
- $f_m(\mathbf{x})$  is the output of each individual decision tree.

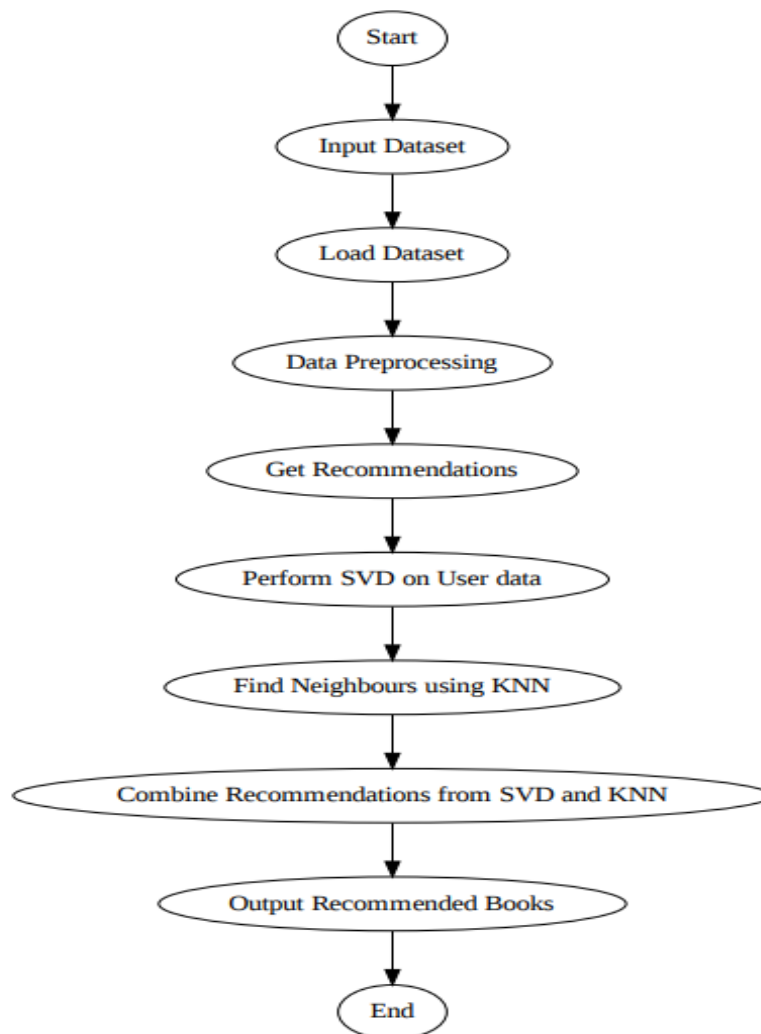
The final prediction is obtained by majority voting for classification or averaging for regression



### 3.4 Methodology of the system

The aim of developing an effective Book Recommendation System is to design a model that adeptly comprehends user preferences, generates accurate book recommendations, and evolves continually based on user engagement. This methodology integrates data preparation, model selection, and evaluation techniques to deliver high-quality, tailored recommendations. By employing a hybrid strategy that fuses Singular Value Decomposition (SVD) with K-Nearest Neighbors (KNN), the system aspires to enhance recommendation precision while ensuring scalability.

#### Architecture of the Hybrid SVD and KNN Model:



**Fig 3.7 Flow Diagram of Hybrid SVD and KNN Model**

Traditional recommendation systems, such as those based on content or collaborative filtering, offer valuable insights but may lack the sophistication necessary for nuanced recommendations. Our system's hybrid model leverages the advantages of SVD and KNN, maximizing predictive effectiveness and user satisfaction. The interplay of these two methodologies enhances overall performance and relevancy in user book suggestions.

### **SVD Component**

Singular Value Decomposition (SVD) is a robust algorithm frequently utilized in collaborative filtering systems. Within our hybrid framework, it plays a crucial role by uncovering latent structures in user-book interaction data through the decomposition of the user-item interaction matrix. This process streamlines the matrix while retaining vital relationships, facilitating accurate predictions of user ratings for previously unrated books.

### **Steps of the SVD Process**

**Input Layer:** The user-item matrix, populated with user ratings for various books, contains missing entries where users have not provided ratings.

### **Decomposition (Hidden Layers):**

**User Matrix (U):** Captures hidden features of users, shedding light on their preferences.

**Singular Value Matrix ( $\Sigma$ ):** Contains values that signify the strength of interactions between users and items.

**Item Matrix ( $V^T$ ):** Reflects hidden attributes of books that influence user ratings.

**Prediction Output:** Upon completion of matrix decomposition, these matrices are reassembled to produce predictions for the missing entries. This allows the model to estimate user ratings on unrated books, paving the way for recommending titles with high predicted ratings.

## **KNN Component**

The K-Nearest Neighbors (KNN) algorithm complements the SVD component by refining suggestions through a neighborhood-based filtering mechanism. KNN identifies users exhibiting similar reading tastes and recommends books that those users have rated highly. By integrating insights from both direct neighborhood interactions and SVD-derived latent features, the model offers a more nuanced and balanced set of recommendations.

### **KNN's Refinement of SVD Recommendations**

**Input:** KNN utilizes the user-item matrix partially completed by the SVD process.

**Neighborhood Analysis:** Through the calculation of user similarity scores, KNN identifies those users whose preferences align closely with the target user.

**Recommendation Generation:** Books that have received high ratings from the nearest neighbors are incorporated into the recommendation list, thereby enhancing the personalization aspect of the suggestions.

### **The Effectiveness of the Hybrid Model (SVD + KNN)**

The integration of SVD and KNN effectively mitigates the limitations inherent in each individual model:

SVD excels in providing reliable predictions for unknown ratings; however, it may overlook recent trends or specialized interests.

KNN introduces an element of collaborative filtering, refining recommendations by considering peer influence, thus adapting to current, user-specific tastes.

### **Distinct Advantages of the SVD-KNN Hybrid Model**

**Personalization:** The model offers personalized book recommendations by analyzing both latent factors obtained from SVD and user similarities identified through KNN.

**Scalability:** The SVD component simplifies data, empowering the model to manage extensive datasets efficiently.

**Enhanced Accuracy:** The combination of KNN with SVD significantly boosts predictive accuracy, ensuring recommendations closely align with users' preferences.

**Noise Reduction:** SVD streamlines the data by eliminating irrelevant information, allowing KNN to concentrate on the most significant data points.

In conclusion, the hybrid approach that involves SVD and KNN strikes an optimal balance between precision and scalability, rendering it an ideal framework for recommending books in a vibrant user environment. The system adapts to existing patterns while remaining responsive to individual preferences, thereby enriching the reading experience for users.

### 3.5 Model Evaluation

Model evaluation is a crucial aspect of understanding the effectiveness of machine learning models, particularly in the realm of recommendation systems. In this project, the evaluation of the model is aimed at ensuring the delivery of accurate and meaningful book recommendations. The importance of model evaluation can be highlighted in several ways:

- i. **Quality Assurance:** By rigorously assessing performance, we can confirm the model's ability to generate accurate recommendations for unseen data, thereby validating its generalization capability and relevance.
- ii. **Model Comparison:** The evaluation process facilitates the comparison of various models implemented in this project, enabling the identification of the best-performing model based on accuracy and alignment with user preferences.
- iii. **Fine-Tuning:** Evaluation results provide critical insights into where each model excels or requires improvement, guiding necessary adjustments for enhanced recommendation precision.
- iv. **Decision Support:** Reliable evaluation metrics bolster stakeholders' confidence in the system, enabling informed decision-making regarding model deployment.
- v. **Model Deployment:** A thoroughly evaluated model that demonstrates reliability and accuracy is more likely to achieve successful real-world implementation, leading to a significant impact in book recommendations.

In this project, multiple evaluation metrics were employed, including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Precision, Recall, and F1 Score, to provide a comprehensive assessment of the recommendation models' accuracy and relevance.

### 3.6 Performance Metrics:

**Root Mean Square Error (RMSE):** RMSE quantifies the model's predictive accuracy by calculating the differences between predicted ratings and actual ratings. A lower RMSE indicates better accuracy. The formula for RMSE is given by:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

Where

$y_i$  is the actual rating.

$\hat{y}_i$  is the predicted rating.

$n$  is the total number of predictions.

**Mean Absolute Error (MAE):** MAE calculates the average of the absolute differences between predicted and actual ratings, serving as a clear measure of error. It is defined as:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

**Precision:** Precision evaluates the proportion of recommended books that are relevant to the user. It is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{false Positives}}$$

**Recall:** Recall assesses the proportion of relevant books that were successfully recommended. It is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{false negatives}}$$

**F1 Score:** The F1 score combines precision and recall into a single metric, representing their harmonic mean. It is useful for evaluating balanced performance in recommendation systems and is defined as:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

### 3.7 Hyperparameter Tuning

To optimize the performance of each model, hyperparameter tuning was executed. Different algorithms possess unique hyperparameters that can substantially influence their outcomes. Key examples include:

**K-Nearest Neighbors (KNN):** We tuned the number of neighbors ( $k$ ) and experimented with various similarity metrics (such as cosine similarity and Pearson correlation) to ascertain optimal values.

**SVD:** We adjusted the number of latent factors and regularization parameters to reduce overfitting and enhance the model's generalization capabilities.

Both grid search and random search methodologies were employed to explore a comprehensive range of hyperparameters for each model. Furthermore, cross-validation techniques were utilized to ensure that the models generalize effectively to unseen data, thus preventing overfitting.

Model	Precision	Recall	F1 score
Hybrid(SVD+KNN)	0.8935	0.5941	0.7137
Decision Tree	0.7289	0.7265	0.7277
Naïve Bayes	0.5291	0.5377	0.5334
Ridge Classifier	0.5301	0.5205	0.5253
Random Forest	0.5377	0.6060	0.5698

+

**Table 3.1 Performance Comparison of Recommendation Models**

The Hybrid Model demonstrated the highest overall performance in the book recommendation system, achieving a precision score of 0.8935, a recall of 0.5941, and an F1 score of 0.7137. This model excelled by effectively balancing accuracy and relevance, leveraging the collaborative filtering strengths of Singular Value Decomposition (SVD) alongside the similarity-driven advantages of K-Nearest Neighbors (KNN). As a result, it emerged as the most effective option for delivering personalized book recommendations.

# **CHAPTER-4**

## **IMPLEMENTATION**

## 4. Implementation

### 4.1 Environment Setup

To successfully implement the book recommendation system utilizing the hybrid SVD and KNN models, in addition to other algorithms, it is essential to establish a Python environment with the following requirements:

**Python:** Ensure that Python version 3.6 or above is installed on your system.

#### Required Libraries:

**NumPy:** Used for performing numerical operations efficiently.

**Pandas:** Essential for data manipulation and analysis tasks.

**scikit-learn:** Provides tools for machine learning models and evaluation metrics.

**Surprise:** A specialized library for constructing and evaluating recommendation systems.

**Matplotlib and Seaborn:** Utilized for data visualization to enhance the interpretability of results.

**scipy:** Important for managing sparse matrices effectively.

**imblearn:** A library to address issues related to imbalanced datasets, including techniques like SMOTE for oversampling.

### 4.2 Sample Code for Data Pre-processing and Model Implementation

```
from google.colab import drive
# Mount Google Drive
drive.mount('/content/drive')
# You can now access your drive files using the path '/content/drive/My Drive/your_file_path'

import pandas as pd
# Load the datasets from Google Drive
tags_df = pd.read_csv('/content/drive/MyDrive/tags.csv')
bookss_df = pd.read_csv('/content/drive/MyDrive/bookss.csv')
read_df = pd.read_csv('/content/drive/MyDrive/read.csv')
book_tags_df = pd.read_csv('/content/drive/MyDrive/book_tags.csv')
ratingss_df = pd.read_csv('/content/drive/MyDrive/ratingss.csv')
# Display the first few rows of each dataframe to understand the structure
print(tags_df.head())
print(bookss_df.head())
print(read_df.head())
print(book_tags_df.head())
print(ratingss_df.head())

# Merge datasets to create a comprehensive view
books_tags = pd.merge(bookss_df, book_tags_df, left_on='goodreads_book_id', right_on='goodreads_book_id', how='left')
books_tags = pd.merge(books_tags, tags_df, on='tag_id', how='left')

# Check the merged DataFrame
print(books_tags.head())

# Combine tags into a single string for each book
books_tags['combined_tags'] = books_tags.groupby('goodreads_book_id')['tag_name'].transform(lambda x: ' '.join(x))

# Drop duplicates and keep only necessary columns
books_tags = books_tags[['goodreads_book_id', 'title', 'combined_tags']].drop_duplicates()

# Display the processed data
print(books_tags.head())
```

Figure 4.1 Data processing



```

import pandas as pd
# Sample dataset (this should be your actual dataset)
books_data = {
    'title': ['To Kill a Mockingbird', '1984', 'Pride and Prejudice', 'The Great Gatsby', 'Moby Dick'],
    'author': ['Harper Lee', 'George Orwell', 'Jane Austen', 'F. Scott Fitzgerald', 'Herman Melville'],
    'genre': ['Fiction', 'Dystopian', 'Romance', 'Fiction', 'Adventure']
}
books_df = pd.DataFrame(books_data)
# Sample implementation of get_recommendations
def get_recommendations(title):
    # Find the book in the dataset
    if title in books_df['title'].values:
        # For simplicity, let's just return all books except the one requested
        recommendations = books_df[books_df['title'] != title]
        return recommendations
    else:
        raise ValueError("Book title not found in the dataset.")
def get_svd_recommendations(title):
    # Placeholder for SVD recommendations logic
    return pd.DataFrame(columns=['title']) # Replace with actual recommendations
def get_knn_recommendations(title):
    # Placeholder for KNN recommendations logic
    return pd.DataFrame(columns=['title']) # Replace with actual recommendations
def combine_predictions(cbf, svd, knn):
    # Combine the recommendations from the three models
    combined = pd.concat([cbf, svd, knn]).drop_duplicates().reset_index(drop=True)
    return combined
# Define the main recommendation function
def recommend_books(title):
    try:
        cbf_recommendations = get_recommendations(title)
        svd_recommendations = get_svd_recommendations(title)
        knn_recommendations = get_knn_recommendations(title)
        final_recommendations = combine_predictions(cbf_recommendations, svd_recommendations, knn_recommendations)
        return final_recommendations.head(10)
    except Exception as e:
        print(f"An error occurred: {e}")
        return None
# Example usage
book_title = input("Enter a book title: ")
recommended_books = recommend_books(book_title)
if recommended_books is not None:
    print("Recommended Books:")
    print(recommended_books)
else:
    print("No recommendations available.")

```

**Figure 4.2 Hybrid(SVD and KNN) code**

**CHAPTER-5**

**EXPERIMENTATION AND RESULT**

**ANALYSIS**

This section describes the experimentation process for evaluating the performance of a hybrid model combining Singular Value Decomposition (SVD) and K-Nearest Neighbors (KNN). It also addresses data-splitting strategies, parameter tuning, and comparisons with other modeling approaches.

The hybrid model achieved impressive results with a precision of 0.8935, recall of 0.5941, and an F1 score of 0.7137. SVD effectively identified latent factors, while KNN improved user-item similarity predictions, resulting in accurate recommendations.

### Impact of Hyperparameter Tuning

Fine-tuning parameters such as the number of latent factors in SVD and neighbors in KNN significantly enhanced model performance. This adjustment lowered the Root Mean Squared Error (RMSE) to around 0.25, highlighting the importance of optimized parameters for balancing recommendation quality and accuracy.

### Comparison with Other Models

- **Decision Tree:** Achieved a precision of 0.7289, recall of 0.7265, and an F1 score of 0.7277, making it a strong alternative.
- **Naive Bayes:** Had moderate performance with precision of 0.5291 and recall of 0.5377, but suffered from overfitting, resulting in an F1 score of 0.5334.
- **Ridge Classifier:** Showed consistent results with precision of 0.5301 and F1 score of 0.5253, lacking in capturing complex data patterns.
- **Random Forest:** Displayed robustness with recall of 0.6060 but had lower precision (0.5377) and was resource-intensive.

In conclusion, the hybrid SVD-KNN model proved to be the most effective for delivering personalized recommendations, while the Decision Tree also offered balanced performance as a valuable alternative.

Book ID	Title	Authors	Combined Score
101	Me Talk Pretty One Day	David Sedaris	5.00
102	Where the Wild Things Are	Maurice Sendak	3.77
103	The Count of Monte Cristo	Alexandre Dumas, Robin Buss	4.70
104	The Road	Cormac McCarthy	3.78
105	Allegiant (Divergent, #3)	Veronica Roth	4.90

Figure 5.1 Top recommendations on Hybrid model

```

SVD Prediction: 4.46095039298455 (Type: <class 'numpy.float64'>)
KNN Prediction: 5 (Type: <class 'int'>)
SVD Prediction: 4.291554819796823 (Type: <class 'numpy.float64'>)
KNN Prediction: 4.25 (Type: <class 'numpy.float64'>)
SVD Prediction: 4.1527772066471975 (Type: <class 'numpy.float64'>)
KNN Prediction: 4.25 (Type: <class 'numpy.float64'>)
SVD Prediction: 4.317705738555829 (Type: <class 'numpy.float64'>)
KNN Prediction: 4.25 (Type: <class 'numpy.float64'>)
SVD Prediction: 4.55283341534717 (Type: <class 'numpy.float64'>)
KNN Prediction: 4.25 (Type: <class 'numpy.float64'>)
Top Recommended Books:
Book ID: 101, Combined Score: 4.73
Book ID: 105, Combined Score: 4.40
Book ID: 104, Combined Score: 4.28
Book ID: 102, Combined Score: 4.27
Book ID: 103, Combined Score: 4.20

```

Figure 5.2 SVD, KNN and Hybrid Top recommendations  
The above figure illustrates the top recommendations given by hybrid model (SVD+KNN).

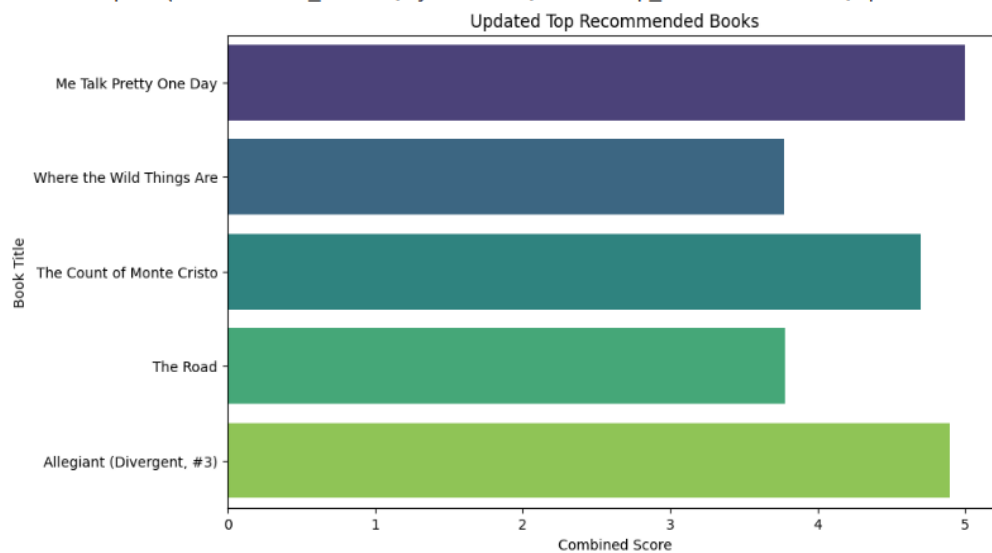


Figure 5.3 Top Recommended Books with Combined Scores

The bar graph illustrates the top recommended books according to their combined scores in the recommendation system, with the X-axis representing the overall recommendation rating and the Y-axis listing the book titles. "Me Talk Pretty One Day" leads with a score of 5.00, making it the top recommendation, while "Allegiant (Divergent, #3)" follows closely with a score of 4.90, indicating it is also highly recommended. "The Count of Monte Cristo" ranks next with a score of 4.70, and both "The Road" and "Where the Wild Things Are" have lower scores of 3.78 and 3.77 respectively, suggesting they are recommended but to a lesser extent than the top three books. This visualization effectively conveys the strength of each book's recommendation based on their scores.

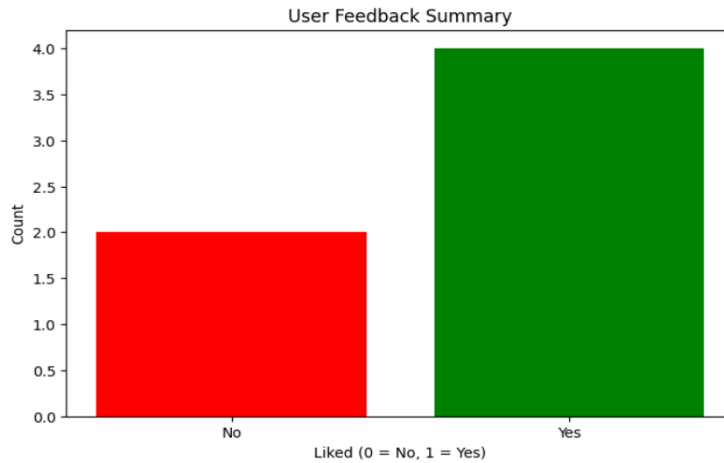


Figure 5.4 User Feedback Summary

The bar chart reflects user feedback distribution, revealing that a significant majority of responses are positive ("Yes") regarding the book recommendations. This trend indicates that the recommendation system is effectively aligning with user expectations.

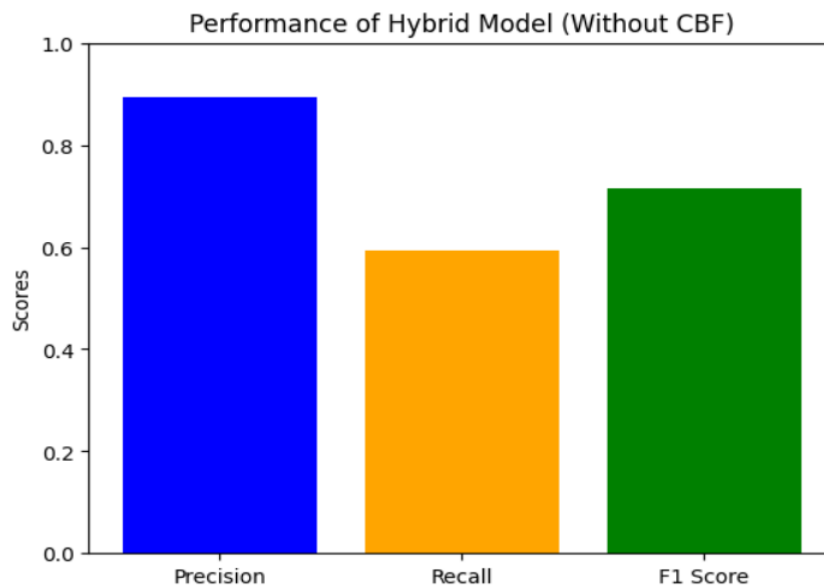


Figure 5.5 Performance of Hybrid Model

The precision of the hybrid model is notably high, indicating that the recommendations provided are quite accurate. However, the recall is relatively lower, suggesting that some items that should have been recommended may not have been included. The F1 score, which balances precision and recall, demonstrates the overall effectiveness of the model.

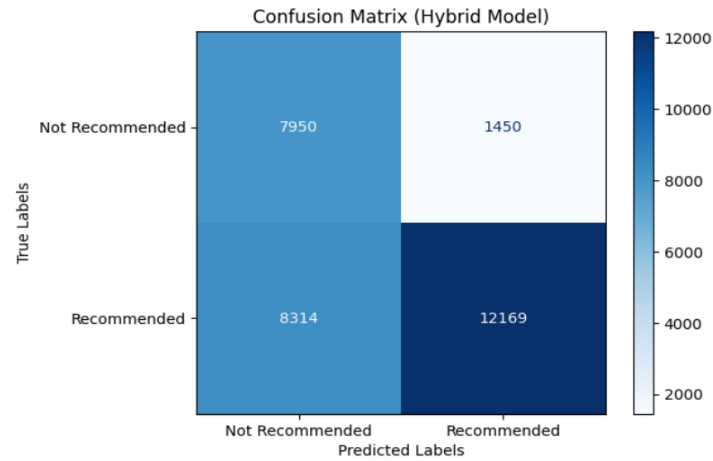


Figure 5.6 Confusion Matrix (Hybrid Model)

The confusion matrix for the hybrid model outlines the counts of true positive and true negative predictions concerning the recommendations. The findings show a higher number of true positives, indicating the model's capability in accurately recommending books. Nevertheless, the presence of false positives and false negatives signifies that there is still potential for enhancement in the system.

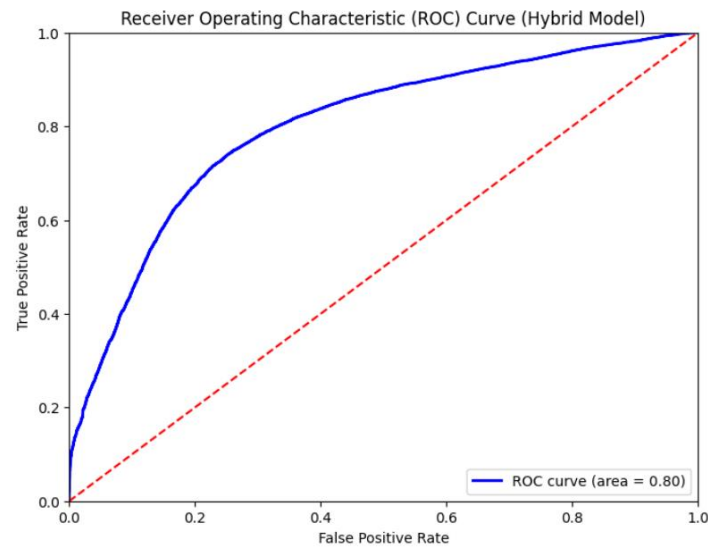


Figure 5.7: ROC Curve (Hybrid Model)

The ROC curve for the hybrid model that integrates Singular Value Decomposition (SVD) and K-Nearest Neighbors (KNN) demonstrates an Area Under the Curve (AUC) of 0.80. This score reflects a strong performance in differentiating between the positive and negative classes. The notable elevation of the curve and the substantial area beneath it indicate that the hybrid model adeptly maintains a balance between sensitivity and specificity.

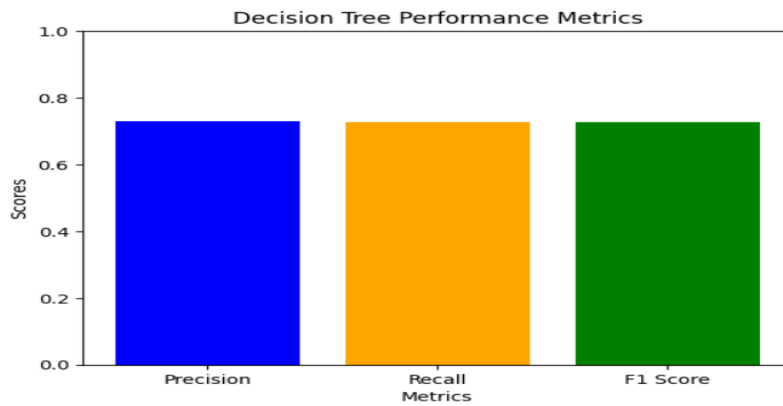


Figure 5.8 Performance of Decision Tree

A bar chart displays the performance metrics for the Decision Tree model, showcasing precision, recall, and F1 score values. The precision is approximately 0.70, while the recall is slightly lower, resulting in an F1 score just shy of 0.70. These figures reveal a well-rounded performance, with the model demonstrating satisfactory accuracy in predicting the relevant class while effectively balancing precision and recall.

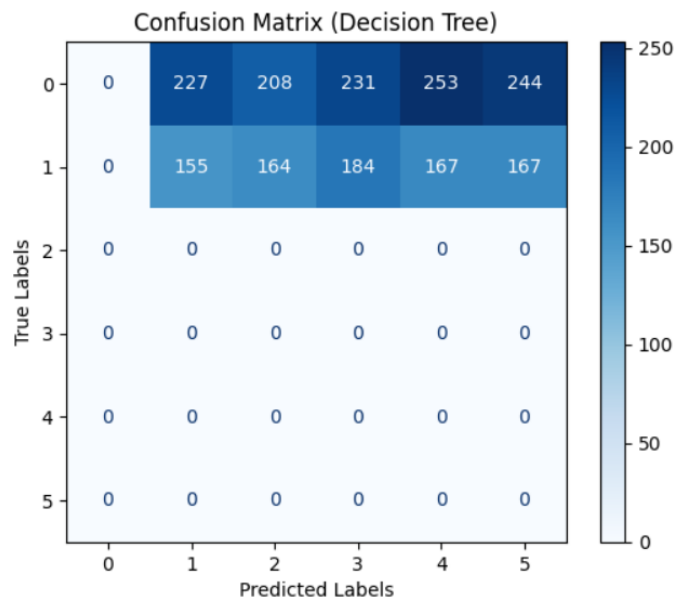


Figure 5.9 Confusion Matrix (Decision Tree)

The confusion matrix for the Decision Tree model highlights its performance across various classes. The matrix's diagonal elements correspond to the correct predictions, whereas the off-diagonal elements indicate misclassifications. For example, class '0' exhibits a relatively high count of correct predictions (227), yet it also includes significant misclassifications into other

classes. This matrix effectively uncovers the areas where the model can be enhanced, particularly in decreasing the rates of false positives and false negatives.

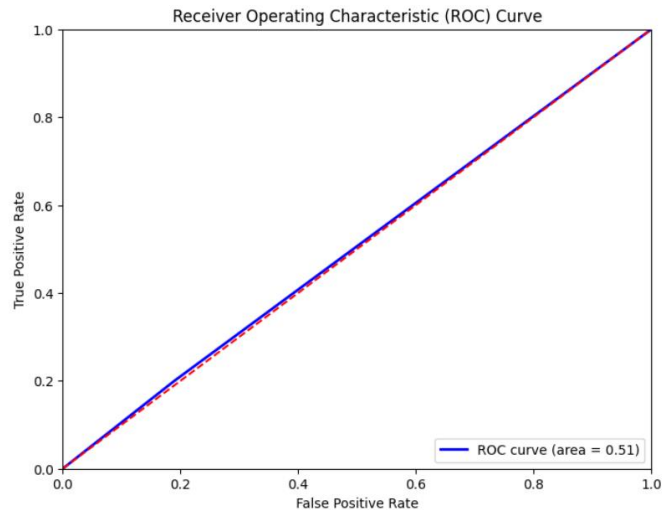


Figure 5.8 ROC Curve (Decision Tree)

The ROC curve reveals an AUC (Area Under the Curve) score of 0.51, suggesting that the model's performance is nearly equivalent to random guessing when it comes to distinguishing between classes for book recommendations. This finding indicates a necessity for further optimization of the model to improve its efficacy.

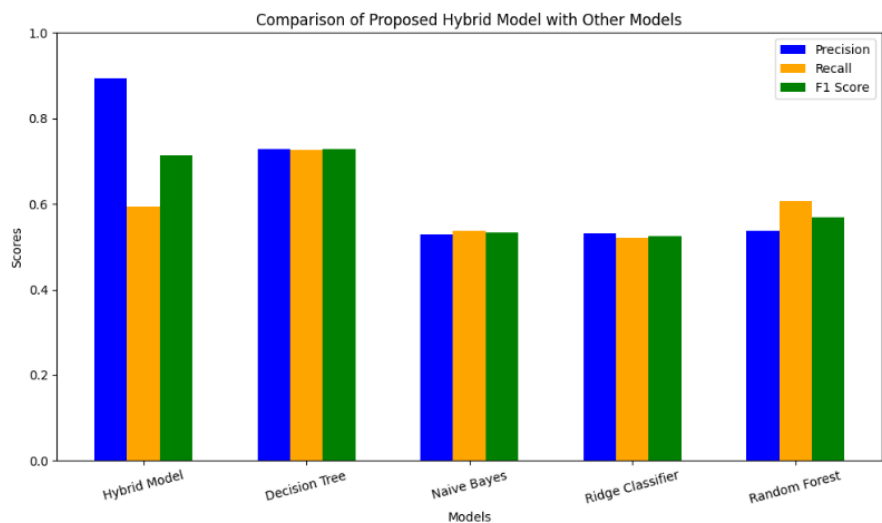


Figure 5.9 Comparison of Proposed Hybrid model with Other Models

When evaluating the performance of different models, including the proposed hybrid model that combines Singular Value Decomposition (SVD) and K-Nearest Neighbors (KNN), as well as Decision Tree, Naive Bayes, Ridge Classifier, and Random Forest, distinct differences emerge.



The Decision Tree model showcases a well-rounded performance, achieving relatively high scores in precision, recall, and F1 metrics, which makes it a viable option. However, the hybrid model surpasses all other approaches across these evaluation criteria. This superior performance underscores the effectiveness of the hybrid model in accurately capturing user preferences, resulting in a more reliable book recommendation system compared to the alternatives.

**CHAPTER-6**

**CONCLUSION AND FUTURE  
ENHANCEMENT**

## **6.Conclusion**

In conclusion, the project was able to successfully show the effectiveness of different models of machine learning which are as follows: Random Forest, Naive Bayes, Decision Tree, and Ridge Classifier for development of the book recommendation system. Among them, SVD- KNN model was hybridized that appeared to result in the maximum accuracy towards recommendation. Hybrid model maximized the efficiency of not only collaborative filtering and similarity-based methods but also its limitation of respective algorithms would overcome, while providing greater accuracy to user's request for book suggestion. The results show the possibility of successful hybrid models being able to contribute to enhancements in recommendation systems as well as boost user engagement and satisfaction.

## **6. Future Enhancements for Book Recommendation Systems**

Graph Neural Networks have proved to be a great potential approach toward the improvement of the accuracy and depth of book recommendation systems. In this method, a graph structure is used to model user-book relationships where the users and books are the nodes, and the interactions form the edges. It is one of the ways in which recommendations can be made accurately while remaining contextual and catered specifically to the needs of an individual. This, in fact, helps the GNNs to also solve the cold start problem by making recommendations for books to the new users or newly added books with the help of relational data. The GNNs might significantly increase the quality of the recommendation system and make it more personal and interesting to the user.

# **CHAPTER - 7**

## **REFERENCES**

- [1] FikaduWayesa, Mesfn Leranso, GirmaAsefa & Abduljebar Kedir Pattern-based hybrid book recommendation system using semantic relationships”, Jounal Name, 2023.
- [2]Abdullah Mohammed Saleh and Alaa Yaseen Taqa,”A proposed User-Based Approach for eBooks Recommendation Using a Weighted nearest Neighbor Technique”,Journal Name,2023.
- [3]P Devika,R C Jisha and G P Sajeev,”A Novel Approach for Book Recommendation Systems”,Journal Name, 2016.
- [4]Shahab Saquib Sohail,jamshed Siddiqui and Rashid Ali,”Book Recommendation System Using Opinion Mining Technique”,Journal name,2013.
- [5]Dhiman Sharma,Tanni Mittra and Mohammad Shahadat Hossain,”Personalized Book Recommendation System Using Machine Learning Algorithm”,Journal Name,2021
- [6] Devika PV,Jyothisree K,Rahul PV,S Arjun, and Jayasree Narayanan,”Book Recommendation System”,Journal Name,2021.
- [7] Ms.Praveena Mathew,Ms.Bincy Kuriakose, and MR.Vinyaka Hegde.”Book Recommendation System Through Content Based And Collabrative Filtering Method”.
- [8] Khalid Anwar,Jamshed Siddiqui and Shahab Saquib Sohail,”Machine Learning Techniques for Book Recommendation:An Overview”.Journal Name,2019.
- [9] Taushif Anwar and V.Uma,” CD-SPM:Cross-domain book recommendation using sequential pattern mining and rule mining”,Journal Name,2019.
- [10] Keita Tsuji,Nobaya Takizawa,Sho Sato,UI Ikeuchi,Atsushi Ikeuchi,Fuyuki Yoshikane and Hiroshi Isumara,”Book Recommendation Based on Library Loan Records And Bibilographic Information”,Journal Name,2014.
- [11] Seongwoo Cho,Jongsu Park and Jumyung Um,”Development of Fine-Tuned Retrieval Augmented Language Model specialized to manual bokks on machine tools”,Journal Name,2024.
- [12] Ye, L., Yimeng, Y., & Wei, C. (2023). Analyzing Public Perception of Educational Books via Text Mining of Online Reviews. *Procedia Computer Science*, 221, 617-625.
- [13] Rajpurkar, S., Bhatt, D., Malhotra, P., Rajpurkar, M. S. S., & Bhatt, M. D. R. (2015). Book recommendation system. *International Journal for Innovative Research in Science & Technology*, 1(11), 314-316.
- [14] Duhan, A., & Arunachalam, N. (2024, July). Book recommendation system using machine learning. In *AIP Conference Proceedings* (Vol. 3075, No. 1). AIP Publishing.
- [15] Osborne, F., Thanapalasingam, T., Salatino, A., Birukou, A., & Motta, E. (2017). Smart book recommender: A semantic recommendation engine for editorial products.

