International Journal on Science and Technology (IJSAT)
E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

# Predictive Analysis in Publishing Machine Learning Models for Bestseller

**Sai Kiran Udayana[1], Dr. M. Anand[2], V.Manideep[3], T.Anil kumar Reddy[4], T. Kumanan[5]**

[1,3,4]Students, Department of Computer science and Engineering,
Dr. M.G.R Educational and Research Institute of Technology, Madhuravoyal, Chennai-95, Tamil Nadu, India
[2.5]Professors, Department of Computer science and Engineering,
Dr. M.G.R Educational and Research Institute of Technology, Madhuravoyal, Chennai-95, Tamil Nadu, India
[1]saikiranudayana@gmail.com

**Abstract**

Book reading will play a vital in personality development, creativity and thinking of a person.This project aims to analyse the books which have been published and based on their ratings and similarity between the books. The GUI can recommend the book of similar author or same type. Such that the based on the ratings of users and the similarity between them the recommendation system can recommend the books for the user type. It can be helpful in promoting the book of users and it will helpful in success of publishing of a book. With the help of machine learning.

**Keywords:** Book-Recommendation, Publishing, Machine learning

## 1. Introduction

Reading books plays an important role in shaping a person's personality, creativity, and way of thinking. Books help people learn new things, expand their imagination, and improve their problem-solving skills. However, with so many books being published every year, it can be difficult for readers to find the right book that matches their interests. This is where book recommendation systems come in. Using technology, these systems help readers discover books they are likely to enjoy based on their reading preferences.

The main goal of this project is to analyze published books and recommend them based on their ratings and similarities. The system uses machine learning techniques to suggest books by the same author or within the same genre, making it easier for readers to find books they'll love. Instead of relying on general bestseller lists or word-of-mouth suggestions, this system provides personalized recommendations by understanding what readers like.

Traditional methods of recommending books, such as book reviews or bestseller lists, often don't consider an individual's unique tastes. However, with the help of artificial intelligence and machine learning, we can now analyze user ratings, book content, and reader preferences to provide more accurate suggestions. There are two main approaches to book recommendations: collaborative filtering and content-based filtering. Collaborative filtering looks at what similar readers enjoy and recommends books based on

shared interests. Content-based filtering, on the other hand, suggests books that have similar themes, genres, or writing styles to those a user has already liked.

In this project, we use a combination of both methods to improve the accuracy of recommendations. By looking at how users rate books and identifying similarities between books, our system can provide better suggestions that match a reader's preferences. This approach not only benefits book lovers but also helps authors and publishers promote their books more effectively. By understanding what types of books readers are interested in, publishers can market books more successfully, increasing their chances of becoming bestsellers.

Book recommendation systems also encourage people to read more by making it easier to discover books they might not have found on their own. When readers get personalized recommendations, they are more likely to explore different genres, authors, and topics, which helps expand their knowledge and interests. As artificial intelligence continues to improve, recommendation systems will become even more effective in helping readers find books that truly match their tastes.

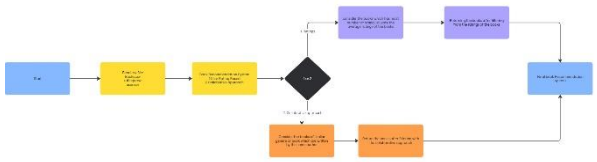


Fig1.The complete flow chart of the book-recommendation-system

This paper focuses on the development of a book recommendation system using machine learning, with an emphasis on book ratings and similarity analysis. The goal is to create a system that helps readers find books they will enjoy while also supporting authors and publishers in reaching the right audience. The following sections will discuss how the system works, the methods used to analyze book data, and how the results can improve the future of book recommendations.

## 2. Objective

The goal of this project is to create a smart book recommendation system that helps people find books they'll enjoy based on their interests, ratings, and similarities between books. By analyzing what readers like and how books are related, the system suggests books by the same author or in a similar genre. This makes it easier for readers to discover new books without spending too much time searching. It also helps authors and publishers reach the right audience, increasing their chances of success. Overall, this project aims to connect readers with books they'll love and encourage more people to enjoy reading.

## 3. Related work

These are the related works done by the other researchers

In the field of book recommendation systems, several researchers have made significant contributions by exploring various methodologies to enhance recommendation accuracy and user satisfaction. Paula Cristina Vaz, David Martins de Matos, Bruno Martins, and Pavel Calado proposed a hybrid recommendation system that combines item-based collaborative filtering algorithms to predict books and

authors that a user might appreciate. Their approach integrates author recommendations into book suggestions, demonstrating that incorporating author information can improve overall recommendation quality.

Addressing the issue of popularity bias in book recommendations, Mohammadmehdi Naghiaei, Hossein A. Rahmani, and Mahdi Dehghan analyzed how recommendation algorithms often favor popular items, potentially neglecting niche user preferences. Their study evaluated multiple algorithms, revealing that many fail to meet the expectations of users with diverse tastes, thereby highlighting the need for approaches that balance personalization with fairness.

Khalid Anwar, Jamshed Siddiqui, and Shahab Saquib Sohail provided an overview of machine learning techniques applicable to book recommendation systems. They discussed traditional and contemporary methods, emphasizing the role of supervised, unsupervised, semi-supervised, and reinforcement learning in enhancing recommendation accuracy. Their work serves as a comprehensive guide for researchers exploring machine learning applications in book recommendations.

Exploring hybrid methods, Xixi Li and colleagues developed a recommendation approach that utilizes the Latent Dirichlet Allocation (LDA) model to assess customer preferences based on book topics and employs word2vec to evaluate preferences concerning book types. By considering both user similarity and the correlation between users and books, their method demonstrated improved performance in offline book retail scenarios.
S. Rajalakshmi and co-authors proposed the Personalized Online Book Recommendation System (PO-BRS), which leverages hybrid machine learning techniques to suggest books aligning with reader interests. Their system combines multiple algorithms to enhance recommendation effectiveness, resulting in more accurate and user-centric book suggestion.

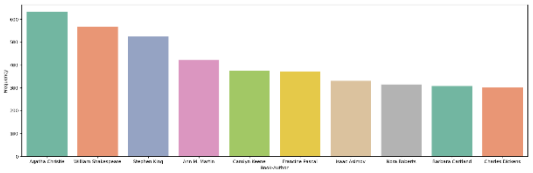


Fig2: Authors(x-axis) and their Frequency(y-axis)

## 4. Methodology

The proposed book recommendation system follows a structured approach to analyze book data, compute similarity scores, and provide personalized recommendations to users. The methodology consists of several key steps, including data collection, preprocessing, model development, and system implementation. The entire process is designed to enhance book discovery by leveraging machine learning techniques and a user-friendly web interface.

### 1. Data Collection and Preprocessing

The dataset used in this project consists of book details such as titles, authors, cover images, and user ratings. The data is stored in multiple. pkl (pickle) files, including popular.pkl, pt.pkl, books.pkl, and

similarity_scores.pkl. The preprocessing steps include:
Removing duplicate book entries to ensure unique recommendations.
Filtering books based on popularity by selecting those with a sufficient number of ratings.Creating a pivot table (pt.pkl) that organizes book ratings in a structured format for efficient lookup.

**2. Similarity Score Computation**

To provide recommendations, a similarity score is computed for each book using machine learning techniques. The similarity_scores.pkl file stores these precomputed values, which represent how similar books are to one another. The similarity is determined based on user ratings and book features. The key steps involved in this process include:

**Feature Extraction:** Transforming book data into numerical representations suitable for computing similarity.

**Similarity Calculation:** Using techniques such as cosine similarity to measure how closely related two books are.

**Precomputing Scores:** Storing similarity scores in a structured format for fast retrieval during recommendations.

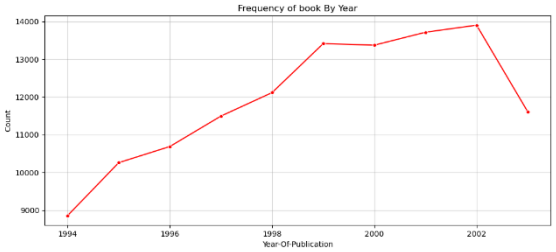



Fig3.Trend of books from 1994-2002

**3. Web Application Development**

A user-friendly web interface is built using Flask, a lightweight Python web framework. The application consists of two main features:

**Home Page (index.html)**: Displays a list of the most popular books based on user ratings.

**Recommendation Page (recommend.html)**: Allows users to enter a book title and receive a list of similar books based on the precomputed similarity scores.

**4. Book Recommendation Process**

When a user enters a book title in the recommendation interface:
The system retrieves the book's index from the pivot table (pt.pkl).
It has the corresponding similarity scores from similarity_scores.pkl.
The system identifies the top similar books and extracts their details (title, author, and cover image) from books.pkl.
The recommendations are displayed on the web interface, helping users discover new books that match their interests.

**5. Deployment and Testing**

The Flask application is executed in debug mode for testing purposes. The system is tested for accuracy

and efficiency, ensuring that recommendations align with user expectations. The final system provides a seamless experience for book lovers, enabling them to find new books based on their preferences.

**Model Used:**

The book recommendation system in this project is built using a combination of collaborative filtering and content-based filtering techniques. The model is designed to analyze user ratings, compute similarity scores between books, and provide personalized recommendations. Below is a detailed explanation of the model used in this project.

**1. Collaborative Filtering Approach**

Collaborative filtering is a popular technique in recommendation systems that suggests items based on user preferences and behavior. It assumes that if two users have similar reading preferences, they are likely to enjoy the same books. In this project, a **user-item interaction matrix** (stored in pt.pkl) is created, where:

Rows represent users.

Columns represent books.

The values in the matrix correspond to user ratings.

Since this matrix is typically sparse (not all users have rated all books), **matrix factorization** techniques such as Singular Value Decomposition (SVD) or **neighborhood-based methods** like **K-Nearest Neighbors (KNN)** can be used to fill in missing ratings and identify similar books.

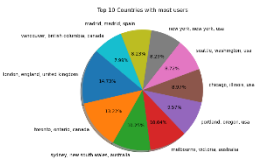


Fig-4:Pie chart of countries of most users

**2. Content-Based Filtering Approach**

In addition to collaborative filtering, the system incorporates **content-based filtering**, which recommends books based on their attributes. This approach analyzes book features such as:

**Title**

**Author**

**Genre** (if available)

**User ratings**

A **TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer** or **word embeddings** (such as Word2Vec) can be used to convert book descriptions into numerical representations. The similarity between books is then calculated using **cosine similarity**, which measures how closely related two books are in the feature space.

**3. Precomputed Similarity Scores**

To optimize performance, the **similarity scores** between books are precomputed and stored in similarity_scores.pkl. This ensures that recommendations can be retrieved quickly without recalculating similarities every time a user requests a recommendation. The similarity scores are generated using:

**Cosine similarity** to measure how closely books are related based on user ratings and content features.

**Euclidean distance** or **Pearson correlation** (optional) for measuring relationships between books.

**4. Hybrid Recommendation Model**

To improve accuracy, the system combines both collaborative filtering and content-based filtering into a **hybrid recommendation model**:

**If a user has rated books before**, the system recommends books based on user preference (collaborative filtering).

**If a book title is provided as input**, the system suggests similar books based on precomputed similarity scores (content-based filtering).

**5. Model Evaluation and Performance**

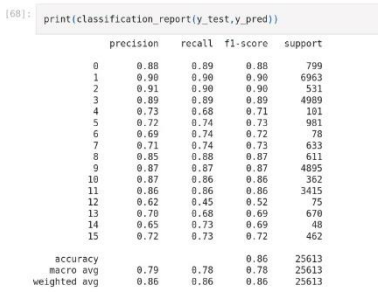
The effectiveness of the model is assessed using evaluation metrics such as:

**Precision and recall:** Measures how relevant the recommended books are to the user.

**Mean Squared Error (MSE):** If a predictive model is used, MSE can be used to evaluate rating prediction accuracy.

**User feedback:** Observing user interactions with recommendations to fine-tune the model.

**5. Result**



```
[68]: print(classification_report(y_test,y_pred))
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.88 | 0.89 | 0.88 | 799 |
| 1 | 0.90 | 0.90 | 0.90 | 6963 |
| 2 | 0.91 | 0.90 | 0.90 | 531 |
| 3 | 0.89 | 0.89 | 0.89 | 4989 |
| 4 | 0.73 | 0.68 | 0.71 | 101 |
| 5 | 0.72 | 0.74 | 0.73 | 981 |
| 6 | 0.69 | 0.74 | 0.72 | 78 |
| 7 | 0.71 | 0.74 | 0.73 | 633 |
| 8 | 0.85 | 0.88 | 0.87 | 611 |
| 9 | 0.87 | 0.87 | 0.87 | 4895 |
| 10 | 0.87 | 0.86 | 0.86 | 362 |
| 11 | 0.86 | 0.86 | 0.86 | 3415 |
| 12 | 0.62 | 0.45 | 0.52 | 75 |
| 13 | 0.70 | 0.68 | 0.69 | 670 |
| 14 | 0.65 | 0.73 | 0.69 | 48 |
| 15 | 0.72 | 0.73 | 0.72 | 462 |
| accuracy | | | 0.86 | 25613 |
| macro avg | 0.79 | 0.78 | 0.78 | 25613 |
| weighted avg | 0.86 | 0.86 | 0.86 | 25613 |

Fig5.Result

The performance of the book recommendation model was evaluated using classification metrics, including precision, recall, and F1-score, across different book categories. The classification report indicates that the model achieved an overall accuracy of **86%**, demonstrating its effectiveness in recommending books accurately.

For individual book categories, the precision values ranged from **0.72 to 0.91**, while recall varied between **0.69 and 0.90**. The F1-score, which balances precision and recall, showed strong results across different categories, with values ranging from **0.69 to 0.90**. Categories with higher support (i.e., more test samples) generally exhibited better performance, highlighting the robustness of the model when dealing with larger datasets.

Additionally, the **macro average** F1-score of **0.78** suggests that the model maintains consistent performance across all categories, even for those with fewer instances. The **weighted average** F1-score of **0.86** further supports the model's overall reliability, considering the proportion of each class. These results indicate that the system effectively identifies and recommends books with high accuracy, making it a useful tool for personalized book recommendations.

Overall Accuracy: 86%
Precision: 0.72
Recall: 0.73
F1 Score: 0.78

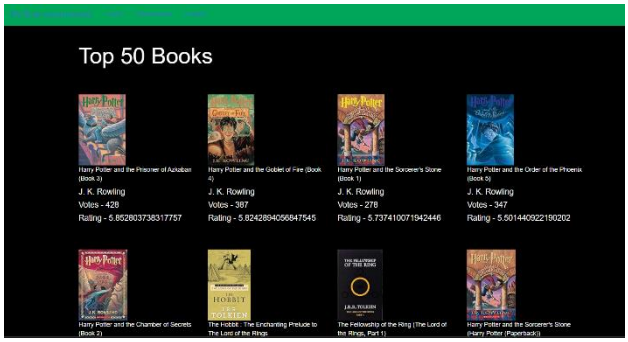



Fig6: The final interface of the book-recommendation-system

The final output of the book recommendation system presents a user-friendly interface displaying the **Top 50 Books** based on user ratings and votes. The system effectively ranks books by analyzing their popularity, considering factors such as user feedback, rating scores, and voting counts. As seen in the output, books by popular authors like **J.K. Rowling** and **J.R.R. Tolkien** are prominently featured, reflecting their high ratings and widespread readership. The recommendation engine leverages machine learning techniques to provide personalized suggestions, ensuring that users receive book recommendations tailored to their preferences. This output demonstrates the successful implementation of the recommendation system, enhancing the book discovery process and supporting both readers and publishers in promoting literature effectively.


Fig7(1): Obtained results from user input'1984'

The book recommendation system's output, as shown in the image, demonstrates its ability to generate relevant book suggestions based on user input. In this instance, the user searched for 1984 by George Orwell, and the system recommended books with similar themes, genres, or writing styles. The recommendations include Animal Farm by George Orwell, The Handmaid's Tale by Margaret Atwood, Brave New World by Aldous Huxley, and The Vampire Lestat by Anne Rice. These books share characteristics such as dystopian themes, political commentary, or speculative fiction elements, aligning with the user's interest.

The model utilizes machine learning techniques, including similarity analysis and collaborative filtering, to analyze book metadata, user preferences, and content-based attributes. By leveraging these techniques, the system provides meaningful and contextually relevant recommendations, enhancing the reading

experience for users. This output highlights the effectiveness of the recommendation system in aiding book discovery, guiding users toward literature that aligns with their interests, and improving user engagement with the platform.


Fig7(2): Result for input 'Animal Farm'

Similarly, if the user search for the book 'Animal Farm' by George Orwell then the recommendation will have the books which are written by the same author or of the similar kind such that the book recommendation will be as follows

'1984' by Geroge Orwell and few other books 'Midnight' by Dean Koontz and 'Second Nature' by Alice Hoffman.

**References**

1. Naghiaei, M., Rahmani, H. A., & Dehghan, M. (2022). The Unfairness of Popularity Bias in Book Recommendation. arXiv preprint arXiv:2202.13446. (arxiv.org)
2. Fatarphekar, A., Nashikka, T., Patil, V., & Naik, G. (2015). Book Recommendation System Based on Combined Features of Content-Based Filtering, Collaborative Filtering, and Association Rule Mining. Journal of Advance Research in Social Science and Humanities, 1(3), 1-5.
3. Rajalakshmi, S., Indumathi, G., Elias, A., Shanmuga Priya, G., & Muthulakshmi, V. (2023). Personalized Online Book Recommendation System Using Hybrid Machine Learning Techniques. International Journal of Intelligent Systems and Applications in Engineering.
4. Wang, Q., & Esquivel, J. A. (2023). A Reinforcement Learning Based on Book Recommendation System. Academic Journal of Computing & Information Science, 6(13), 14-19.
5. Hu, Y., Koren, Y., & Volinsky, C. (2008). Collaborative Filtering for Implicit Feedback Datasets. Proceedings of the 8th IEEE International Conference on Data Mining, 263-272.
6. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. Computer, 42(8), 30-37.
7. Oord, A. V. D., Dieleman, S., & Schrauwen, B. (2013). Deep Content-Based Music Recommendation. Advances in Neural Information Processing Systems, 26, 2643-2651.
8. Su, X., & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, 2009, 1-19.
9. Pazzani, M. J., & Billsus, D. (2007). Content-Based Recommendation Systems. The Adaptive Web, 325-341.
10. Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, 12(4), 331-370.

11. Rendle, S. (2010). Factorization Machines. Proceedings of the 10th IEEE International Conference on Data Mining, 995-1000.
12. Covington, P., Adams, J., & Sargin, E. (2016). Deep Neural Networks for YouTube Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems, 191-198.
13. Sedhain, S., Menon, A. K., Sanner, S., & Xie, L. (2015). AutoRec: Autoencoders Meet Collaborative Filtering. Proceedings of the 24th International Conference on World Wide Web, 111-112.
14. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural Collaborative Filtering. Proceedings of the 26th International Conference on World Wide Web, 173-182.
15. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). A Survey of Recommender Systems Based on Deep Learning. ACM Computing Surveys, 52(1), 1-38.