International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

Bookshelf: A Book Recommendation System Using Collaborative Filtering

Nancy Taswala, Anish Kamble, Sana Khan

Abstract – In the huge amount of literature, finding the perfect book to please your literary thirst can be a task. With countless genres, authors, and styles to choose from, it's easy to be lost in a sea of literacy. Fortunately, we live where technology and data can come to our rescue, guiding us toward our next great read. In this paper, we present the algorithms used to create Bookshelf and how we tried several algorithms before using collaborative filtering. Next, we discussed how the whole system is designed. The objective of a book recommender system is to create a personalized and engaging experience for book lovers, by providing them with relevant and interesting book recommendations that match their unique preferences and interests.
Overall, the objective of bookshelf is to create a win-win situation by providing users with personalized recommendations that enhance their reading experience.

***Key Words*: Recommender System, Collaborative Filtering, Cosine Similarity.**

1. INTRODUCTION

Fortunately, we live where technology and data can come to our rescue, guiding us towards our next great read. Welcome to the world of book recommendation systems, where Bookshelf join forces with the timeless joy of reading to help you select your next literary gem. Books have been a source of knowledge, inspiration, and entertainment for centuries, offering windows into different worlds, perspectives, and emotions. Whether you're an avid reader, a casual book enthusiast, or someone just starting their reading journey, the quest for the perfect book is a universal experience. It's a journey filled with excitement, curiosity & adventure where the destination remains a mystery until you turn the first page. This book recommendation system, or "bookish guide," is designed to be your trusted companion on this literary adventure. It's not just about algorithms and data; it's about harnessing the power of technology to enhance your reading experience and connect you with books that resonate with your unique tastes and preferences Along the way.

we'll introduce you to the different types of book recommendation systems by collaborative filtering, but this journey isn't just about the mechanics of recommendation algorithms.

It's about the profound connection between books and readers. We'll explore the role of how technology is shaping the future of libraries and book stores. Users will provide reviews and ratings for books they have read, helping others to discover new titles or make informed decisions. Bookshelf will provide information about authors, including information about previous books, personal information and link to other books.

With so many books available online, users may feel overwhelmed and uncertain about which books to choose. Bookshelf help users navigate this information overload by providing tailored recommendations that match their interests and needs. Users may not be aware of all the books that could interest them, and may miss out on potential favorites. Bookshelf will help users broaden their awareness by suggesting books they may not have discovered otherwise. Generic book recommendations may not be relevant or useful for all users, as they do not take into account individual preferences, tastes, and reading habits. Bookshelf will provide personalized recommendations that match the user's unique profile and behavior
.
Bookshelf will include a recommendation engine that analyses user data, such as their reading history and preferences, and provides personalized recommends.

It will also include a search function that allows users to search for books based on various criteria, such as title, author, genre and ratings. A rating system that allows users to rate books they have read, which can be used to improve the accuracy of future recommendations.

Bookshelf will be fast and responsive, with recommendations generated in real-time or near-real-time. It is designed in such a way that it can handle large volume of users and books with robust security to protect user data, including authentication, authorization, and encryption.

This paper will provide an in-depth examination of the workings of Bookshelf, as well as explore potential areas for improvement. Furthermore, the paper will scrutinize the package's internal architecture and how it determines the correct rule to activate based on the user's input. The paper will also outline the trials and errors we encountered in developing the package and how we arrived at its current state.

2. LITERATURE SURVEY

A recommendation system generally depends upon the inputs of a user and their relationship between the products. To fully understand the workflow of a book recommendation system, a thorough review on equivalent systems has been done as a guideline for development of the proposed system and overcoming the weaknesses which present in the existing system. In the work, we have directly reviewed several book recommendation websites including Goodreads, TBR, epic reads, etc.

Goodreads:
Goodreads is a popular social cataloguing and book recommendation platform that allows users to discover, review, and discuss books. While it offers several useful features, it also has its disadvantages like privacy concerns, less intuitive, limited customization, inconsistent data quality.

TBR:
The acronym “TBR” stands for “to be read”, it’s a term commonly used by readers to refer to the list of books they intend to read in the future. This platform allows users to catalogue books, user reviews, connection with other users, book information. While it offers several useful features, it also has its disadvantages like privacy & security, data import challenges, no information about author, technical errors.

Epic reads:
Epic Reads is a popular online community and platform dedicated to young adult literature. It provides book recommendations, reviews, author interviews, and various interactive features for young adult book enthusiasts. While it offers several useful features, it also has its disadvantages like content quality control, privacy concerns, instability of website, lack of genres.

3. COLLABORATIVE FILTERING

The collaborative filtering algorithm uses user's behavior to recommend items. They exploit the behavior of other users and factors of tracking history, ratings, and choices. Other users' behavior and book preferences are used to recommend books to new users [1]. However, it can be difficult to include secondary functionality, i.e. functions beyond the query or ISBN. For book recommendations, extras may include genre and rating. The inclusion of available extra features improves the quality of the model.
So, in bookshelf, we decided to use collaborative filtering (CF). It is a technique used to make personalized recommendations by analyzing the preferences and interactions of other users. Collaborative filtering can introduce users to new books or authors they may not have discovered otherwise [4]. By, leveraging the preferences of similar users, the system can recommend items that a user might find interesting but hasn’t come across yet. This can involve techniques like calculating similarities based on shared preferences or using more sophisticated methods like matrix factorization.

It can mitigate the cold start problem, which occurs when a new user joins the Bookshelf. It will provide user with trending books and will also provide top authors page where user can easily find top authors and books written by the top authors.

It can scale well with large datasets and user bases easily. As the number of books and users grows, the system can still provide relevant recommendations by efficiently processing user-item interaction data.

Also, CF can adapt to changes in user preferences over time. As users interact with more books and their tastes evolve, the system can continually update its recommendations to reflect these changes by storing their last search history.

$$\text{Simil}(x,y) = \frac{\sum_i \sum_{Ixy} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt[2]{\sum_i \sum_{Ixy} (r_{x,i} - \bar{r_x})} \sqrt[2]{\sum_i \sum_{Ixy} (r_{x,i} - \bar{r_y})}}$$

Collaborative Filtering Formula

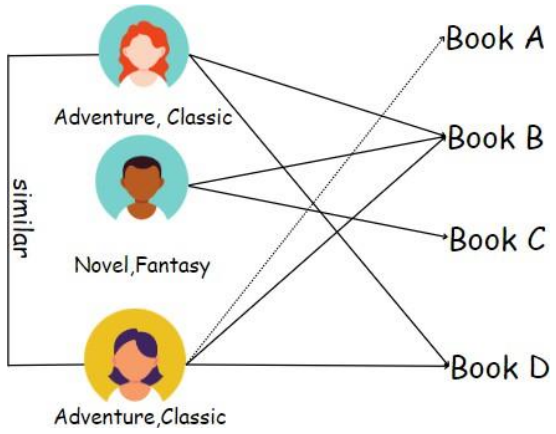


Fig 1: Visual Representation of Collaborative Filtering

User A prefers book genres A, B, C, and user C prefers to read books B, D, hence we can conclude that the likings of user A and user C are very similar. Since user A likes book D as well, we can deduce that the user A may also like book D, therefore book D would be recommended to the user The general idea of the algorithm is based on a review of previous ratings provided by the users. Find the neighbor user as alpha who exhibits similar interest with target user beta, and then suggests the items which the neighbor user alpha preferred to target user beta, the predicted score which the target user may give on the item is obtained by the score calculation of neighbor user alpha on the item.

4. COSINE SIMILARITY

Cosine similarity is a measure of similarity between two sequences, in our cases, these sequences will be tables. It is versatile and widely used metric in collaborative system-based recommendation systems [7]. In sparse user-item interaction matrix where not all users have rated all books, cosine similarity provides an effective way to measure similarity even without complete data.

It allows the system to identify patterns and relationships between users and item based on available ratings [8].

User-Item Matrix:

The first step is to represent the user-item interactions in a matrix. Each row represents a user, each column represents an item (book), and the entries represent the user's rating or interaction with the item.

Profile Creation:

For each user, a profile vector is created representing the user's preferences or characteristics. This vector typically contains ratings given by the user to items.

Similarity Calculation:

Cosine similarity is then used to measure the similarity between user profiles. Cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space. In the context of recommendation systems, it calculates the cosine of the angle between the profile vectors of two users. The formula for cosine similarity between the profile vectors of two users. The formula for cosine similarity between two vectors u and v is:

$$\text{Similarity(u,v)} = \frac{u.v}{\|u\|\|v\|}$$

Where u.v denotes the dot product of vectors u and v, and $\|u\|$ *and* $\|v\|$ denote the Euclidean norms of vectors u and v respectively.

Neighborhood Selection:

Once the similarity scores between users are computed, a set of similar users (neighborhood) is selected for each user. This can be done by selecting the k users with the highest similarity scores to the target user.

Rating Prediction:

For items that a user has not interacted with, ratings are predicted based on the ratings of similar users. One common approach is to compute a weighted average of the ratings of the similar users, where the weights are the similarity scores. The predicted rating r_{ui} for user u and item i can be calculated as:

$$r_{u\,i} = \frac{\sum_{v \in N(u)} similarity\ (u,v) * r_{vi}}{\sum_{v \in N(u)} |similarity(u,v)|}$$

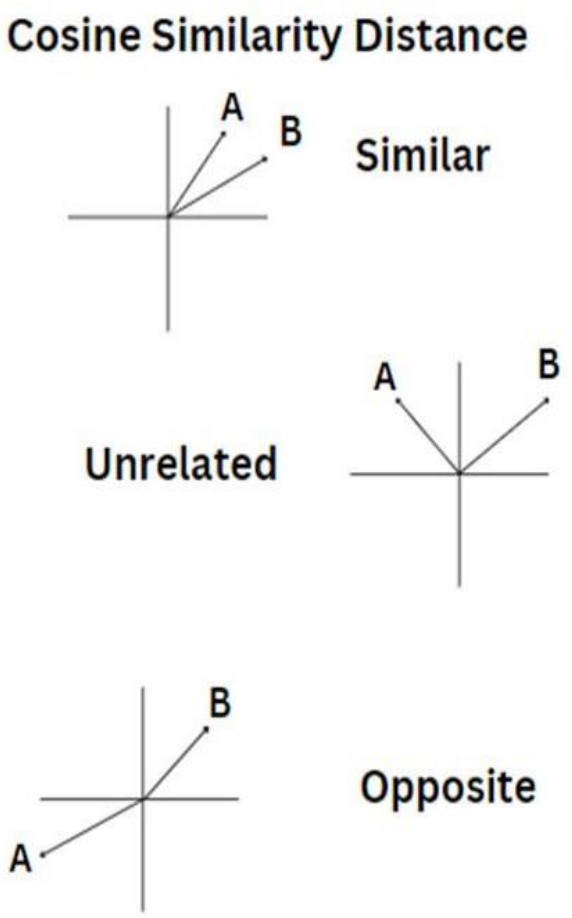
Where N(u) represents the set of similar users to user u,r_{vi} is the rating of item i by user v, and $|similarity(u,v)|$ denotes the absolute value of the similarity between users u and v.

Recommendation Generation:

Finally, recommendations are generated by selecting the top n items with the highest predicted ratings for the user.

$$\text{Similairity}=\cos(\theta)\frac{A.B}{\|A\|\|B\|}=\frac{\sum_{i=0}^{n} Ai\,Bi}{\sqrt{\sum_{i=0}^{n} Ai^2}\sqrt{\sum_{i=0}^{n} Bi^2}}$$

Cosine Similarity Formula



5. Architecture & Workflow:

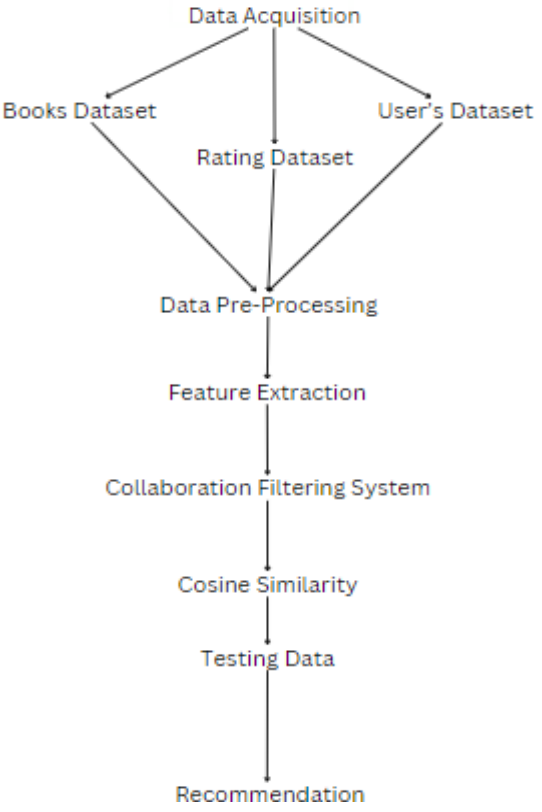


Fig 2: Architecture & Workflow of Bookshelf

Bookshelf uses collaborative filtering and cosine similarity approach to provide personal customization to users. The first approach is where the previously introduced algorithms come into play. For a recommendation, Bookshelf iterates through each rule and its cases, for each case it finds similarity between tables. The image below nicely explains how this works in a graphical manner. this diagram a visual representation of Bookshelf that maps out the physical implementation for components of a software system. It shows how "data acquisition" is divided into 3 sub parts of it and gathered in the end for "data pre-processing" which is further connected to "feature extraction" and further to "collaboration filtering", where we are specifying which algorithms we have used in Bookshelf. The next step will be "testing data" which in result gives recommendations.

7. TOOLS USED:

SOFTWARE REQUIREMENTS

Operating System:
The software is compatible with Windows 10 or higher versions. This ensures that the book recommendation system can run seamlessly on modern Windows environments, taking advantage of the latest features and optimizations.

Programming Language:
The backend of the system is developed using the Python programming language. Python 3.9 or a later version is required, providing a robust and versatile foundation for implementing the voice assistant's functionalities.

Chrome Driver:
Chrome Driver is specified as part of the software requirements. This component is essential for certain web-related functionalities, ensuring compatibility with the Chrome web browser.

Database Management System:
The database management of this book recommendation system will be reliable and scalable using MySQL.

Design:
The frontend of book recommendation system is built by using popular software such as Figma for enhancing the user experience.

HARDWARE REQUIREMENTS

Space Requirements:
The system requires a minimum of 2 MB of available space. This ensures that the necessary files and dependencies can be accommodated without overwhelming the storage capacity.

Internet Connection:
An internet connection is essential for certain functionalities, such as genre searches and accessing recommendations. This requirement allows the book recommendation to stay updated and retrieve real-time information.

RAM Requirements:
A minimum of 4GB of RAM is specified. This ensures smooth execution and responsiveness of the book recommendation, especially when handling complex tasks and processing large datasets.

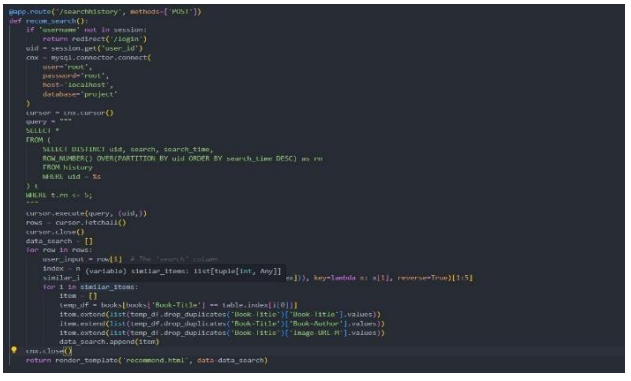
Pen Drive or External HDD:
A pen drive or external hard disk drive is recommended for safe backup purposes. This facilitates the secure storage of essential data and project files, offering a precautionary measure against data loss.

8. IMPLEMENTATION OF MODULES:

8.1 Module 1:
MAIN RECOMMENDER MODULE

In the book recommendation systems, collaborative filtering stands out as a powerful approach. This method leverages collective user behavior and preferences to make personalized recommendations. By analyzing past interactions, collaborative filtering identifies patterns and similarities among users. This enables the system to suggest books that align with a user's tastes and interests. Bookshelf will let users discover new literary gems and delve into genres they might not have explored otherwise, enhancing their reading experience.

8.2 Module 2:
RECOMMENDATION BASED ON SEARCH HISTORY



Bookshelf leverages a user's search history holding immense potential for enhancing the recommendation page. By analyzing the user's past searches, Bookshelf can identify patterns, preferences, and interests, enabling it to suggest books that align closely with the user's tastes.

Collaborative filtering further enriches this process by considering similarities between users with similar search histories, thus offering recommendations based on collective preferences. By integrating these insights into the recommendation page, users are presented with a personalized selection of books tailored to their individual preferences, fostering a more engaging and satisfying browsing experience.

8.3 Module 3:
GENRE FILTRATION

```
@app.route('/genre', methods=['GET', 'POST'])
def genre():
    # Load your data
    with open('C:\\Codes\\VS Code\\Bookshelf\\p_trend.pkl','rb') as f:
        data_genre = pickle.load(f)

    books = None
    if request.method == 'POST':
        genre = request.form.get('genre')
        filtered_books = data_genre[data_genre['Genre'] == genre]
        sorted_books = filtered_books.sort_values('AvgRatings', ascending=False)
        books = sorted_books.to_dict('records')
        default_image_url = 'https://m.media-amazon.com/images/I/41mnOFwAAaL._AC_UF1000,1000_QL80_.jpg'
        for book in books:
            if not book['Image-URL-N']:
                book['Image-URL-N'] = default_image_url
    genres = data_genre['Genre'].unique()
    return render_template('genre.html', genres=genres, books=books)
```

In Book Recommendation system, genre filtration plays a pivotal role in enhancing user satisfaction and engagement. By leveraging collaborative filtering techniques, Bookshelf can analyze user behavior and preferences to generate personalized recommendations. Introducing genre filtration adds an additional layer of customization, allowing users to refine their recommendations based on specific interests and tastes. This approach not only streamlines the browsing experience by presenting users with books tailored to their preferred genres but also enhances the system's accuracy in predicting user preferences. As a result, users are presented with a curated selection of books with their unique reading preferences, fostering a more enjoyable and personalized browsing experience.

8.4 Module 4:
TOP AUTHORS

```
@app.route('/authors', methods=['GET'])
def authors():

    # Load your data
    with open('C:\\Codes\\VS Code\\Bookshelf\\p_trend.pkl','rb') as f:
        data_author = pickle.load(f)

    # Get the top 10 authors
    top_authors = data_author['Book-Author'].value_counts().nlargest(10).index.tolist()

    # Prepare the data for the template
    authors = [{'name': author, 'image_url': author_images[author]} for author in top_authors]

    return render_template('authors.html', authors=authors)
```

In the Bookshelf, the "Top Authors" page serves as a pivotal feature, leveraging the collective wisdom of users to highlight authors whose works resonate widely within the community. The "Top Authors" page thus becomes a dynamic showcase of literary talent, offering users a curated selection of prolific writers whose books are likely to captivate and inspire based on the collective preferences of the community. This personalized approach enhances user engagement and satisfaction, fostering a sense of discovery and connection within the reading community.

8.5 Module 5:
BOOKS BY TOP-AUTHORS

```
@app.route('/author/<author_name>', methods=['GET'])
def author_books(author_name):
    # Load your data
    with open('C:\\Codes\\VS Code\\Bookshelf\\p_trend.pkl','rb') as f:
        data_author = pickle.load(f)

    # Filter the books by the selected author
    books = data_author[data_author['Book-Author'] == author_name]

    filtered_books = data_author[data_author['Book-Author'] == author_name]

    books = filtered_books.to_dict('records')
    return render_template('author_books.html', author_name=author_name, books=books)
if __name__ == '__main__':
    app.run(debug=True)
```

In Bookshelf, showcasing top authors and their related books plays a pivotal role in enhancing user engagement and satisfaction. By highlighting top authors, Bookshelf leverages the popularity and influence of these writers to guide users towards quality content aligned with their preferences. Collaborative filtering utilizes user behavior data to identify patterns and similarities among users with similar tastes, enabling the system to recommend books authored by those favored writers. Additionally, displaying books related to the authors allows for a curated browsing experience, where users can explore a diverse range of titles within their preferred author's genre or style, further enriching their reading journey. Overall, integrating top authors and their related books into the recommendation system enhances personalization, engagement, and satisfaction, ultimately optimizing the user experience.

8.6 Module 6:
TRENDING BOOKS

Displaying a top 50 trending books page within a book recommendation system involves harnessing the collective wisdom of users to identify popular and engaging reads. This approach not only promotes discovery of new and noteworthy titles but also enhances user engagement by providing a curated selection tailored to their tastes and preferences. Additionally, displaying the top 50 trending books fosters a sense of community by showcasing what others are currently enjoying, thereby encouraging further exploration.

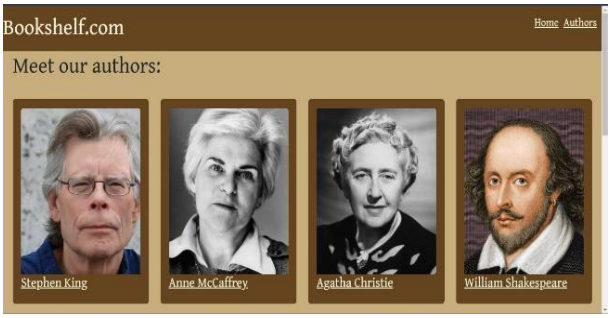
9. RESULT / OUTCOME

MAIN RECOMMENDER MODULE

SEARCH BY GENRE

TRENDING BOOKS

TOP AUTHORS



AUTHORS RELATED BOOKS

10. CONCLUSION

In conclusion, embarking on the development journey of a python-based Book Recommendation System utilizing cutting-edge technology presents an exciting and valuable opportunity. This project holds the potential to significantly enhance productivity while offering users a practical and efficient interface for various tasks.

Developers can harness the capabilities of Bookshelf with wide range of functionalities, such as search by genres, ratings and authors. Also, bookshelf provide recommendations for your next read and by providing a list of top trending books and authors.

The synergy between Python, Figma and the evolving landscape of functionalities presents a fertile ground for innovation and expansion in the realm of personalized recommendations. With the powerful combination of python's versatility and Figma's cutting -edge capabilities, there's many opportunities for developers to push the boundaries of what is possible in creating intuitive and customized recommendations.

11. REFERENCES:

I. IEEE Xplore https://ieeexplore.ieee.org

II. Hindawi : Research paper on book recommendation algorithm https://www.hindawi.com

III. ResearchGate:OnlineBook recommendation system https://www.researchgate.net

IV. IJCRT: Book recommendation system using machine learning https://ijcrt.org

V. IJRASET: Book recommendation system using machine learning https://www.ijraset.com

VI. IOPscience: Online book recommendation system using collaborative filtering https://iopscience.iop.org

VII. Academia.edu : a survey on book recommendation systems https://www.academia.edu

VIII. Semantic Scholar : Personalized book recommendation system https://www.semanticscholar.org

IX. SSRN library: Enhanced book recommendation system https://papers.ssrn.com

X. Research Square: book recommendation system using machine learning https://researchsquare.com

XI. Learning to Recommend with Social Trust Ensemble: https://researchsquare.com

XII. Neighborhood-Based Collaborative Filtering Methods: https://encora.com

XIII. Slope one Predictors for Online Rating-Based Collaborative Filtering: https://arXiv.com

XIV. Deep Learning for Recommender Systems: https://arXiv.com

XV. Factorization Machines for Recommender System: https://towardsdatascience.com