Book Recommender System

STA5073Z Assignment 1

Tinotenda Muzambi (MZMTIN002)

Introduction

In the age of information overload, recommender systems have become indispensable tools for helping users navigate vast catalogs of products and content. This report details the development of a book recommender system that leverages multiple approaches to provide recommendations. Our system combines user-based collaborative filtering (UBCF), item-based collaborative filtering (IBCF), matrix factorization (MF), and an ensemble model to create a robust recommendation engine.

User-Based Collaborative Filtering

UBCF operates on the principle that users with similar reading preferences in the past are likely to have similar preferences in the future. This approach identifies users with similar reading histories and recommends books that these similar users have enjoyed but the target user hasn't yet read.[1]

The motivation behind UBCF is its ability to capture complex user preferences that may not be easily describable by item features alone. It can uncover unexpected recommendations based on the collective wisdom of similar users, potentially leading to serendipitous discoveries.[2]

Item-Based Collaborative Filtering

Item-based collaborative filtering (IBCF) focuses on the relationships between items rather than users. It assumes that users will prefer items similar to those they've liked in the past. IBCF calculates similarity between books based on user rating patterns and recommends books similar to those the user has already rated highly.[3]

The primary motivation for IBCF is its scalability and stability compared to UBCF, especially in systems where the number of items is significantly smaller than the number of users. It's also less affected by the entry and exit of users from the system, making it more robust to user churn.[2]

Matrix Factorisation

Matrix factorization is a latent factor model that aims to uncover hidden features that explain observed rating patterns. It works by decomposing the user-item interaction matrix into lower-dimensional user and item matrices. These lower-dimensional representations can capture nuanced aspects of user preferences and item characteristics that aren't explicitly modeled.[4]

The motivation for including matrix factorization is its ability to handle sparsity in the rating matrix and its capacity to generalize to unseen user-item interactions. It often outperforms memory-based approaches (like UBCF and IBCF) in terms of accuracy and can provide recommendations even when explicit rating data is limited.[5]

The Data

The data is a partially preprocessed version of the ”Book-Crossing” dataset consisting of three tables: Ratings, Books, and Users. The Ratings table contains the ratings given by users to books. The Books table contains information about the books, such as the title and year of publication. The Users table contains information about the users, such as their age. The data can be downloaded here from Kaggle. It contains 278 858 users providing 1 149 780 ratings about 271 379 books.

A description of the features in each table is provided below:

Ratings Table

Table 1: Ratings Table Features

Feature Name	Description	Data Type
User.ID	Unique identifier for the user	Integer
ISBN	Unique identifier for the book	String
Book.Rating	Rating given by the user to the book	Integer

Books Table

Table 2: Books Table Features

Feature Name	Description	Data Type
ISBN	Unique identifier for the book	String

Feature Name	Description	Data Type
Book.Title	Title of the book	String
Book.Author	Author of the book	String
Year.Of.Publication	Year the book was published	Integer
Publisher	Publisher of the book	String
Image.URL.S	URL of the book cover in small size	String
Image.URL.M	URL of the book cover in medium size	String
Image.URL.L	URL of the book cover in large size	String

Users Table

Table 3: Users Table Features

Feature Name	Description	Data Type
User.ID	Unique identifier for the user	Integer
Location	Location of the user	String
Age	Age of the user	Integer

Data Pre-processing

We will merge the Books, Ratings and User datasets into a single dataframe. This will make working with the data easier. Due to the large size of the data, we will then perform various functions to reduce the number of observations.

This will serve the purposes of both reducing the size of the data and ensuring that we have enough data to build accurate models. This is important as collaborative filtering models require a certain amount of data to make accurate predictions.

We will also perform some data cleaning. Part of which includes recoding the user and book IDs to be sequential integers starting from 0. This is necessary for the `recosystem` package which will be used in the matrix factorisation model.

Exploratory Data Analysis

We perform some exploratory data analysis to understand the data better by looking at the structure and summary of the data. We also look at the distribution of the ratings, age, and year of publication to understand the data better.

Looking at the structure and summary of the data, we identify some cleaning that needs to be done. We will do the following to clean up the data:

- Remove the columns that we do not need.
- Replace ages less than 13 and greater than 100 with NA.
- Convert the data types of the columns to the appropriate types.
- Replace year of publication with NA where it is before 1900.
- Remove any rows with missing values.

We will look at the distribution of the ratings, age, and year of publication to understand the data better. We will use histograms to visualise the distributions.

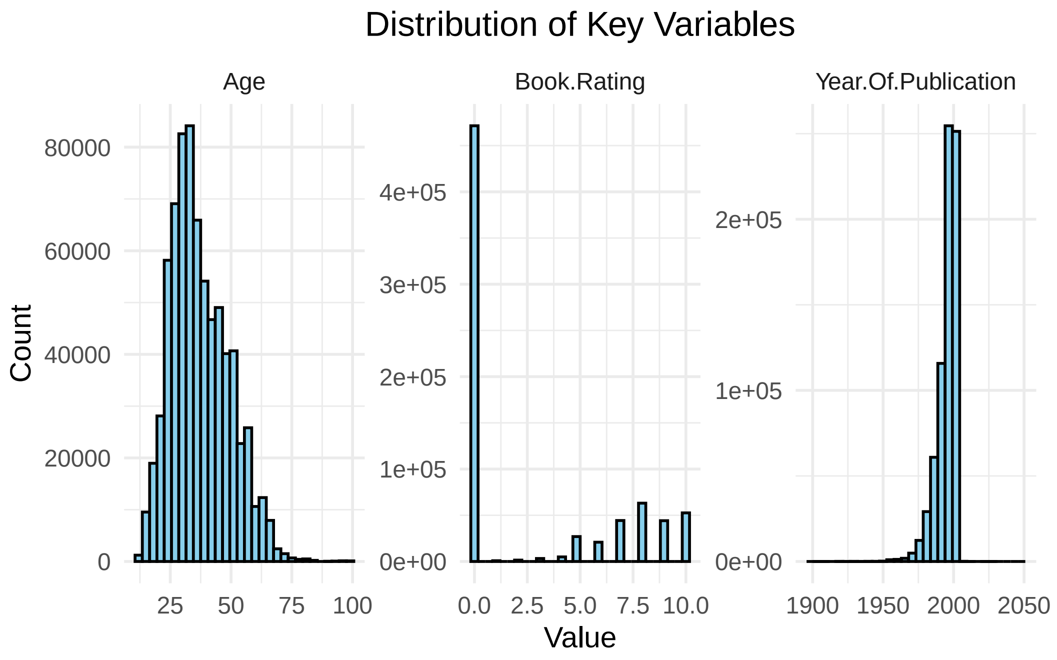

Figure 1: Distribution of Key Variables

Figure 1 is a faceted histogram which shows the distributions. For the age, the distribution is right-skewed with most users being between 20 and 40 years old. This is expected as younger users are more likely to use online platforms to rate books. For the year of publication, the distribution is left-skewed with most books being published between 1990 and 2000. This is expected as older books are less likely to be rated. For the ratings, the distribution is right-skewed with most ratings being between 5 and 10. We also observe a large number of zero ratings. Two possible explanations for this could be that a majority of the users posting reviews are doing so because they had a bad experience or the site the data is derived from could be configured to automatically input a 0 for the book rating. We will drop the zero ratings to improve the accuracy of the recommendation systems.

Before building the recommendation systems, we will drop users and books with less than five ratings and only keep the columns we need for the analysis. Namely, the user ID, ISBN, book rating and the book title. We will also then take a sample of 35 000 ratings for computational efficiency.

We will now look at the distribution of the ratings after removing the zero ratings. Figure 2 depicts the distribution of the ratings. It shows that most of the ratings are between 5 and 10. This is expected as users are more likely to rate books that they enjoyed.

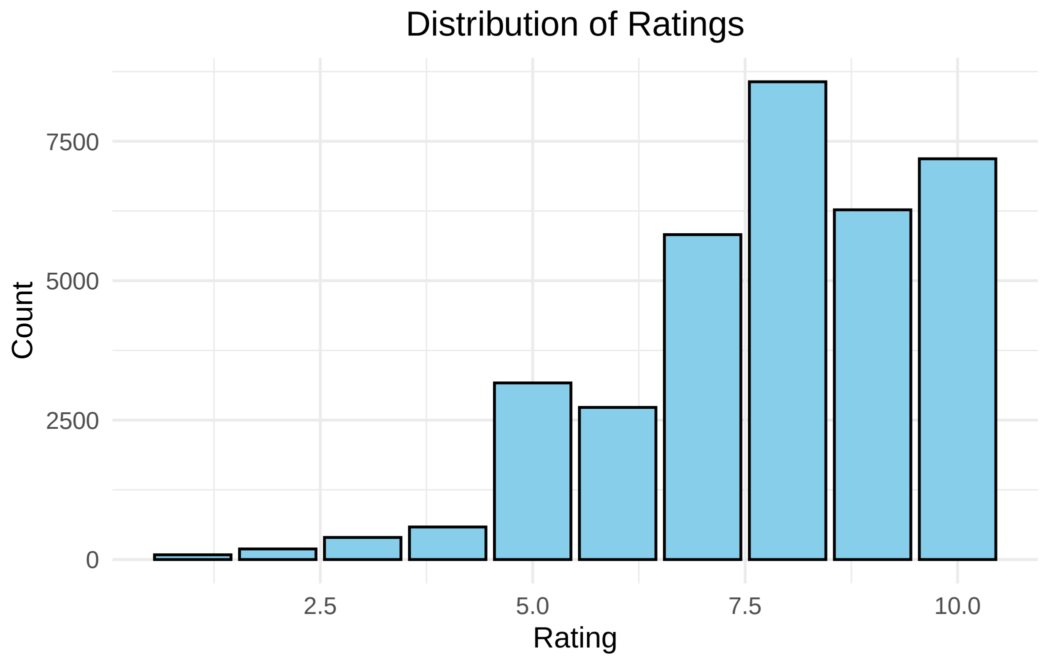

Figure 2: Distribution of Ratings

Recommendation Systems

We begin building our recommendation systems by building a UBCF system. We then build an IBCF system. Finally, we build a MF system before creating an ensemble model that combines the three models.

User-Based Collaborative Filtering

We start by building a UBCF system. User based collaborative filtering finds users with similar consumption patterns as yourself and gives you the content that these similar users found interesting.[6] How it works is that it first creates a user-item matrix. The user-item matrix is a matrix where the rows represent users, the columns represent items, and the cells represent the ratings given by users to items. This matrix is then used to calculate the cosine similarity between users. We then use the similarity matrix to predict the ratings for the books.

Item-Based Collaborative Filtering

We then build an IBCF system. IBCF uses similarity between the items to determine whether a user would like it or not. The same matrix created in the UBCF system is used. This matrix is used to calculate the cosine similarity between items. We then use the similarity matrix to predict the ratings for the books.

Matrix Factorisation

We then build a matrix factorisation system using recosystem. Matrix factorization is an extensively used technique in collaborative filtering recommendation systems. Its objective is to factorise a user-item matrix into two low-ranked matrices, the user-factor matrix and the item-factor matrix, that can predict new items that users might be interested in. This is achieved in by multiplying the two factor matrices.[7]

We first allocate training and test splits before setting up the model. We also perform some hyperparameter tuning to find the best values for the learning rate and dim. We then make predictions on the test set in order to evaluate the accuracy of the model using Root Mean Square Error(RMSE). The root mean square error (RMSE) measures the average difference between a statistical model's predicted values and the actual values.[8] We then apply regularisation to the model and evaluate the accuracy again.

Hyperparameter Tuning

The initial grid for the hyperparameter search involved varying `dim` from 0 - 50 and `lrate` from 0.01 - 0.1. From the initial search the optimal parameters were `dim` = 50 and `lrate` = 0.1. Since these values were on the edge of the grid, we performed another search with `dim` varying from 50 - 100 and `lrate` from 0.1 - 0.5. The optimal parameters found were `dim` = 75 and `lrate` = 0.1. The model is then retrained with these optimal hyperparameters.

Model Evaluation

With the models set up, we can do some testing. We start by creating the ratings matrix. Predictions are then made on the test set to then evaluate the accuracy of the matrix factorisation model using RMSE. We then apply regularisation to the model and evaluate the accuracy again.

The RMSE of the matrix factorisation model without regularisation was foundt to be 1.84. This tells us that on average, the model's predictions are off by 1.84 units. We now apply regularisation to the model and evaluate the accuracy again.

Regularisation

With regularisation, the RMSE of the matrix factorisation model is 1.84. This tells us that on average, the model's predictions are off by 1.84 units. Compared to the RMSE of the matrix factorisation model without regularisation, the regularised model is approximately the same. This could be due to the fact that the model was already performing well without regularisation. Regularisation is used to prevent overfitting and improve the generalisation of the model. In this case, the model was already generalising well without regularisation.

Ensemble Model

We now create an ensemble model that combines the UBCF, IBCF, and MF models. We then evaluate the accuracy of the ensemble model using RMSE. The ensemble model will be created by simply averaging the predictions of the three models.

The RMSE of the ensemble model is 2.93. This tells us that on average, the model's predictions are off by 2.93 units. Compared to the RMSE of the matrix factorisation model of 1.84, the ensemble model is less accurate. This could be due to the fact that the ensemble model is a simple average of the three models. More sophisticated ensemble methods could be used to improve the accuracy of the ensemble model. Namely, methods that take into account the performance of the individual models and assign weights to the models based on their performance. This would allow the ensemble model to take advantage of the strengths of the individual models and improve the accuracy of the predictions. However, the ensemble model is still useful as it combines the strengths of the UBCF, IBCF, and MF models and provides a more robust recommendation engine.

Recommendations

We present a function that recommends books to a user based on the ratings matrix, the models, and the combined ratings data. The function takes a user ID and the number of recommendations as input and returns the top recommended books for the user. We test it based on a much smaller dataset to see how it performs by recommending 5 books to user 38206

Looking at the user 38206's previous ratings, we can see that they have rated 5 books as shown in the table below:

User ID	Book Title	Rating
38206	Beach Roses	9
38206	Secrets of the Heart	8
38206	Blacklist: A V.I. Warshawski Novel	10
38206	Serpent's Dance	9
38206	Catching Midnight	9

The function recommends the following books to user 38206:

- Angela's Ashes
- Harry Potter and the Chamber of Secrets (Book 2)
- Angels & Demons
- Dragonfly in Amber
- Harry Potter and the Sorcerer's Stone (Book 1)

We prompted ChatGPT to provide a summary of the recommendations. The generated response indicated that books like Angels & Demons and Angela's Ashes seem like the most fitting recommendations based on the user's previous ratings. Dragonfly in Amber could also work, but the two Harry Potter books might feel too light or fantastical for their typical preferences, though they could still appreciate them if they enjoy more variety in their reading.[9] This summary provides a good overview of the recommendations and highlights the similarities and differences between the recommended books and the user's previous ratings. This tells us that our recommendation system is able to provide fairly accurate recommendations based on the user's previous ratings. It should be noted however that the recommendations are based on a small sample of the data and may not be representative of the user's actual preferences.

Conclusion

In this project, we built a book recommendation system using collaborative filtering and matrix factorisation. We started by loading the data and performing some data pre-processing. We then built a user-based collaborative filtering system, an item-based collaborative filtering system, and a matrix factorisation system. We evaluated the accuracy of the models using RMSE. Finally, we created an ensemble model that combines the user-based collaborative filtering, item-based collaborative filtering, and matrix factorisation models. The ensemble model was evaluated using RMSE. The results showed that the matrix factorisation model had a lower RMSE compared to the ensemble model. The ensemble model is a simple average of the three models and could be improved by using more sophisticated ensemble methods. Overall, the book recommendation system provides a robust recommendation engine that combines the strengths of the UBCF, IBCF, and MF models.

The system is able to provide accurate recommendations based on the user's previous ratings and can help users discover new books that they may enjoy. The system can be further improved by using more sophisticated ensemble methods and by incorporating additional features such as user demographics and book genres. This would allow the system to provide more personalised recommendations and improve the overall user experience.

References

1. Su, X. (2009). *A survey of collaborative filtering techniques.*
2. Ricci, F., Rokach, L., & Shapira, B. (2010). Introduction to recommender systems handbook. In *Recommender systems handbook* (pp. 1–35). Springer.
3. Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web*, 285–295.
4. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37.
5. Aggarwal, C. C. et al. (2016). *Recommender systems* (Vol. 1). Springer.
6. Boström, P., & Filipsson, M. (2017). *Comparison of user based and item based collaborative filtering recommendation services.*
7. Isinkaye, F. O. (2023). Matrix factorization in recommender systems: Algorithms, applications, and peculiar challenges. *IETE Journal of Research*, *69*(9), 6087–6100.
8. Frost, J. *Root Mean Square Error (RMSE) — statisticsbyjim.com.* https://statisticsbyjim.com/regression/root-mean-square-error-rmse/.
9. OpenAI. (2024). *ChatGPT.* https://chatgpt.com.