Book Recommendation System

Lavesh Jain, Chetan Nain,Vamsidhar reddy Menthem
San Jose State University
May 2022

***Abstract*-- Recommender systems were originally defined as ones in which "people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients" (Resnick & Varian 1997). The term now has a broader connotation, describing any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options. Recommender Systems give the user a limited collection of things that are well-matched to the description from a big variety of goods and a description of the user's demands. A book recommendation system, on the other hand, gives a degree of comfort and personalization that allows the user to connect more and read books that best suit their needs.**

***Index Terms*— Data Mining, Machine Learning, Recommender Systems.**

I. Introduction

In today's world where the number of options available for one product is increasing and the attention span of a user is decreasing It becomes extremely difficult to retain a customer. Customer retention is key to boosting revenue. A 5% increase in customer retention can increase company revenue by 25-95%. This is where recommender systems come into the picture. Companies like Netflix invest heavily in their recommender systems to show relevant content to the user.

People often wondered which book to read next. Books recommendation is yet another application of the ubiquitous recommender system present all around us. These systems affect our decisions and determine what food we would like to order, which songs to listen to, which movies to watch, and even with whom we can connect on social media. The capacity of a recommender system to be comprehensive and relevant is dependent on the effective extraction of hidden patterns from the data. In this paper, book information is retrieved using data from websites like Goodreads and Google Books.

This project is aimed to target the interest of book readers and recommend relevant books to readers. Book reading doesn't come easy to everyone. Most people are very peculiar about the books they read. This is well justified as it isn't as lucid as it looks. Two books can have similar storylines but have different writing styles or two books by the same author can have similar writing styles but an entirely different plot. Two books can be the same but published in two different years or one can be the latest edition of the older one with some modifications.

We propose to build and train the recommender system model on the books data set. Recommender systems are generally divided into two main categories: collaborative filtering and content-based systems. We propose to utilize both these methods and build a hybrid recommendation system.

Dataset: The data has been collected from web using the Goodreads and Google books API. We extracted data of about 10k books and stored it in books_data as csv format. It includes features like book_id, ISBN number, title, author, ratings etc. Apart from this we extracted data for ratings as book_ratings_data and book tags as book_tags_data.

Books Meta Data:

Feature Name	Type	Description
id	Numerical	Id for the dataset
book_id	Numerical	Id of each book
best_book_id	Numerical	Id of the latest version
work_id	Numerical	Abstract ID for the book
books_count	Numerical	Number of Edition
isbn	Numerical	The International Standard Book Number.
isbn13	Numerical	13 digit ISBN
authors	Text	Name of the Authors
original_publication_year	Numerical	Year of Publication of First Edition
original_title	Numerical	Title of Publication of First Edition
title	Text	Title of the Book
language_code	Text	Language the book is published in
average_rating	Numerical	Average ratings of the book
ratings_count	Numerical	Total counts of ratings given
work_ratings_count	Numerical	Total number of text reviews
work_text_reviews_count	Numerical	Total text reviews
ratings_1	Numerical	Total number of 1 ratings given
ratings_2	Numerical	Total number of 2 ratings given
ratings_3	Numerical	Total number of 3 ratings given
ratings_4	Numerical	Total number of 4 ratings given
ratings_5	Numerical	Total number of 5 ratings given

II. Data Preparation

The process of converting raw data into a comprehensible format is known as data preparation. We can't deal with raw data; thus, this is a key stage in data mining. Before using machine learning or data mining methods, make sure the data is of good quality.

1. Data Preprocessing:

Data preparation is the process of transforming raw data into an understandable format. We can't work with raw data, thus this is an important step in the data mining process. Make sure the data is of excellent quality before applying machine learning or data mining technologies. Data preparation is used to guarantee that the data is of high quality. Quality may be determined using the following criteria:

In preprocessing we are doing five major steps:

A. Checking for null values and replacing with suitable values.
B. Detecting Outliers
C. Organizing the data by getting it into proper format so that it can be used for analysis
D. Summary generation using google books api.
E. Clearing the dataset so that it can be used for model preparation.

2. Feature Selection:

When creating a predictive model, feature selection is the process of minimizing the number of input variables. It is preferable to limit the number of input variables in order to reduce modeling computational costs and, in certain situations, increase model performance. In this process we too did some feature selection by removing original_title,isbn and few other redundant features.

3. Data Visualization:

A data visualization is a graphical representation of information and data.Visualizing data using charts, graphs, and maps makes it easier to see and understand trends, outliers, and patterns in data.

3.1 Top Rated Books:

Observation: The below plot helps us to visualize 15 top rated books. Maximum rating of about 4.85 .There is a clear visual representation of top rated books according to the average ratings provided on X-axis, which ranges from 0-5 only.

fig 3.1 Top Rated Books

3.2 Best 15 books by Average rating:

Observation: In the below bar graph, the bars are proportional to the values they represent, and the data is visualized using sns barplots. Average score has been taken on the X-axis and book title has been taken on the Y-axis. A visual representation of the bars is created by comparing both Average Score and book titles.

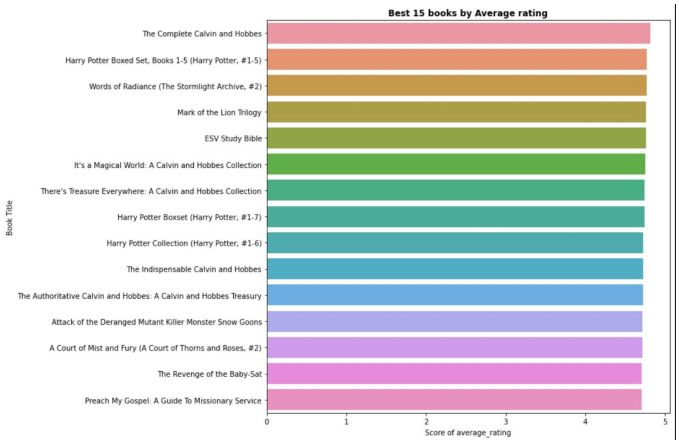


fig 2.2 Best 15 books by Average rating

3.3 Authors with Most Books:

Observation: In the below bar graph, the bars are proportional to the values they represent, and the data is visualized using sns barplots. Number of Books has been taken on the X-axis and the Author has been taken on the Y-axis. A visual representation of the bars is created by comparing both Number of books and Authors.

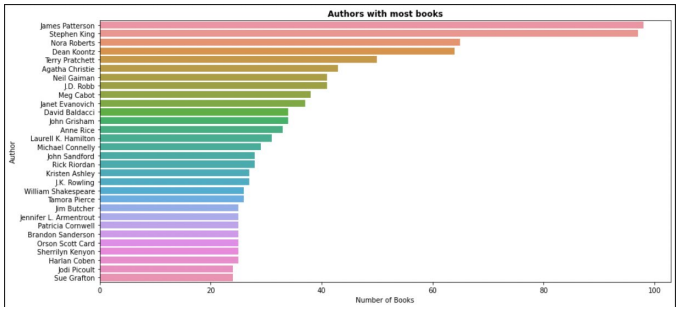


fig 2.3 Authors with Most Books

3.4 Top Authors:

Observation: Interactive Barplot between count and the authors and visualized top authors of books in the dataset in the form of bar graph.

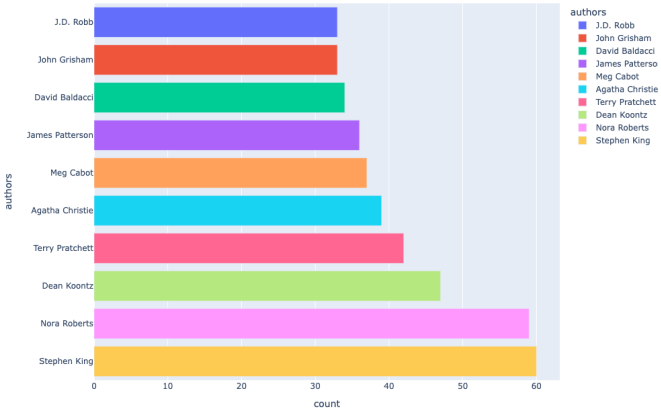
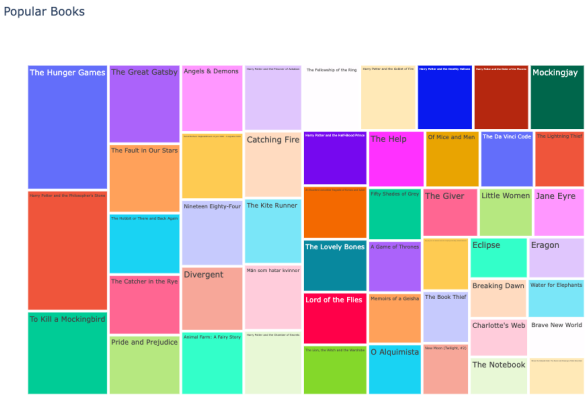


fig 2.4 Top Authors:

Treemap for top authors:



3.5 Percentage of Ratings According to Authors:

Interactive Barplot between Percentage of Ratings and Authors and visualized the Percentage of Ratings According to Authors.

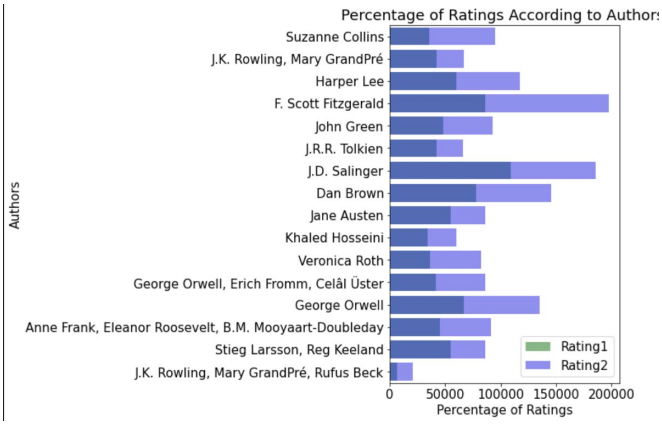


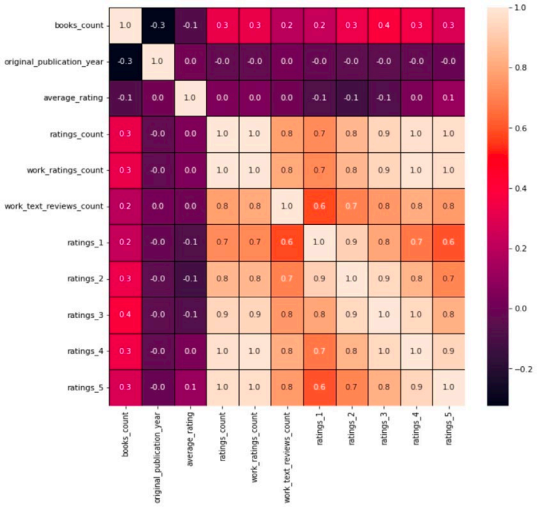
fig 3.5 Percentage of Ratings According to Authors

4. Feature Engineering:

Feature engineering is the process of selecting and transforming the most important variables from raw data when building a predictive model with machine learning or statistical modeling. The purpose of feature engineering and selection is to increase machine learning (ML) algorithm performance. In our dataset too, we had book tags data available with us but still we opted for extracting genres from google api for better accuracy.

5. Correlation analysis:

Correlation analysis: Correlation analysis is a statistical tool used in research to determine the strength of a linear relationship between two variables and compute their association. Simply defined, correlation analysis computes the amount of change in one variable as a result of a change in the other. Here we have plotted a heatmap trying to identify the correlation between all the variables. This way we can easily identify the strong relationship between various features.



III. METHODS

After detailed processing of the data. Next, comes the process of using this data to bring useful information out of it. We tried different models in both the major recommendation techniques i.e. content-based: where the item description matters and collaborative filtering: which works on the similarity between two users and items. We explored clustering techniques to group similar items like K nearest neighbors,

1. **Collaborative-**

Collaborative methods for recommender systems are methods that are based solely on the past interactions recorded between users and items in order to produce new recommendations. These interactions are stored in the so-called "user-item interactions matrix"

Fig. 3.1. User item interaction. Ex: ratings given by users to books.

The main idea that rules collaborative methods is that these past user-item interactions are sufficient to detect similar users and/or similar items and make predictions based on these estimated proximities.

The class of collaborative filtering algorithms is divided into two sub-categories that are generally called memory based and model based approaches. Memory based approaches directly works with values of recorded interactions, assuming no model, and are essentially based on nearest neighbors search. Model based approaches assume an underlying "generative" model that explains the user-item interactions and try to discover it in order to make new predictions.

Collaborative filtering is divided into two parts:

User-User

In order to make a new recommendation to a user, the user-user method roughly tries to identify users with the most similar "interactions profile" (nearest neighbors) in order to suggest items that are the most popular among these neighbors.

For making recommendations to our readers. We represented their interactions (ratings) with different books in vectors. Then, we computed the "similarity" between our user of interest and every other user. That similarity measure is such that two users with similar interactions on the same book should be considered as being close. Once similarities to every users have been computed, we can keep the K-nearest-neighbors to our user and then suggest the most popular items among them.

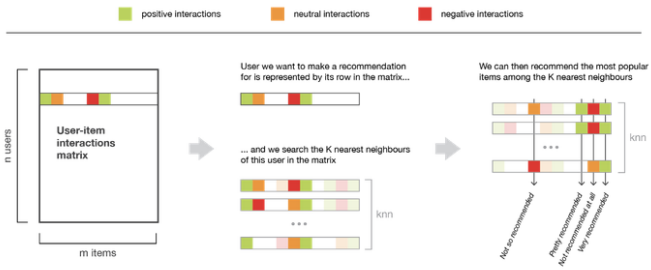


Fig. 3.2. Illustration of User-User method.

Item-Item

The idea of item-item method is to find items similar to the ones the user already "positively" interacted with. Two items are considered to be similar if most of the users that have interacted with both of them did it in a similar way.

We considered the item, user liked the most and represented it by its vector of interaction with every users. Then, we computed similarities between the "best item" and all the other items. Once the similarities have been computed, we kept the K-nearest-neighbors to the selected "best item" that are new to our user of interest and recommend these items.

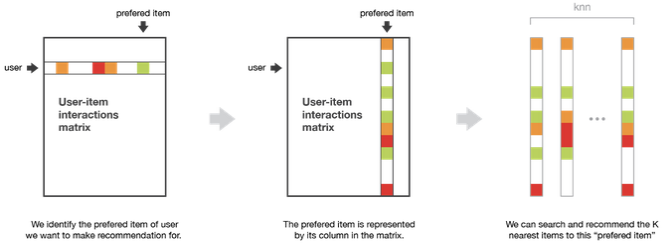


Fig. 3.3. Illustration of Item-Item method.

Collaborative Filtering Using K-Nearest Neighbors (KNN):

Collaborative filtering based on k-nearest neighbors (KNN) KNN is a machine learning algorithm that finds clusters of similar users based on their book ratings and makes predictions based on the average of the top (KNN) k-nearest neighbors K-nearest neighbors (KNN) is a simple, supervised machine learning algorithm that can be used for both classification and regression. This method is easy to implement and understand, but it becomes significantly slow as the size of the data in use increases. k is a positive integer N (Nearest) N (Neighbors)

2. Content-based Filtering-

Content based approaches use additional information about users and/or items. The idea of content based methods is to try to build a model, based on the available "features", that explain the observed user-item interactions. Content based methods suffer far less from the cold start problem than collaborative approaches: new users or items can be described by their characteristics (content) and so relevant suggestions can be done for these new entities. The idea of content based methods is to try to build a model, based on the available "features", that explain the observed user-item interactions.

Content based methods suffer far less from the cold start problem than collaborative approaches: new users or items can be described by their characteristics (content) and so relevant suggestions can be done for these new entities.

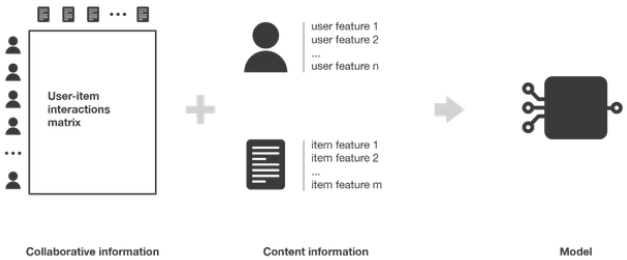


Fig. 3.4.Content based methods paradigm.

The methodology we adopted for the content based approach is user-centric. In this for each user we want to train a simple linear regression that takes item features as inputs and output the rating for this item.We denote M the user-item interaction matrix, we stack into a matrix X row vectors representing users coefficients to be learned and we stack into a matrix Y row vectors representing items features that are given. Then, for a given user i, we learn the coefficients in X_i by solving the following optimisation problem:

$$X_i = \underset{X_i}{argmin} \frac{1}{2} \sum_{(i,j)\in E} [(X_i)(Y_j)^T - M_{ij}]^2 + \frac{\lambda}{2}(\sum_k (X_{ik})^2)$$

Where i is fixed.

We used SVD to train and predict user rating for unread books based on the ratings of the other books that user has rated.

Model-2: (Using Cosine similarity)

For content filtering, cosine similarity is used to map attributes (in my case, text) to determine which items are most similar.As a result, there is no way to measure the accuracy of the models, and the results are more subjective.

Tfidf and Count Vectorization:

Term frequency-inverse document frequency (tfidf) or count vectorization is used to extract features. Based on cosine similarity, both count and tfidf seem viable, but tfidf may be more accurate because it is better at recommending the same authors and series.

tfidf vectorizer:

I chose a book and then calculated the similarity between my chosen book and the rest of the books. From there the recommendation system produced output for both tfidf and count vectorization models.

```
def get_books_recommendations(title, cosine_sim=cosine_sim_author):
    idx = indices[title]

    # Get the pairwsie similarity scores of all books with that book
    sim_score = list(enumerate(cosine_sim_author[idx]))

    # Sort the books based on the similarity scores
    sim_score = sorted(sim_score, key=lambda x: x[1], reverse=True)

    # Get the scores of the 10 most similar books
    sim_score = sim_score[1:11]

    # Get the book indices
    book_indices = [i[0] for i in sim_score]

    # Return the top 10 most similar books
    return list(content_data['original_title'].iloc[book_indices])
```

For the model Tfidf :

```
def author_bookshows(book):
    for book in book:
        print(book)
```

```
vamsi_books1 = get_books_recommendations('The Hobbit', cosine_sim_author)
author_bookshows(vamsi_books1)
```

```
The Hobbit or There and Back Again
 The Fellowship of the Ring
The Two Towers
The Return of the King
The Lord of the Rings
The Hobbit and The Lord of the Rings
Unfinished Tales of Númenor and Middle-Earth
Nikola Tesla: Imagination and the Man That Invented the 20th Century
Entwined
The Children of Húrin
```

3. Hybrid-

It is evident that both the techniques are equally good and complement each other. We explored and studied recommendation technologies based on content filtering and user collaborative filtering and propose a hybrid recommendation algorithm based on content and user collaborative filtering. This method not only makes use of the advantages of content filtering but also can carry out similarity matching filtering for all items, especially when the items are not evaluated by any user, which can be filtered out and recommended to users, thus avoiding the problem of early level.

At the same time, this method also takes advantage of the advantages of collaborative filtering. When the number of users and evaluation levels are large, the user rating data matrix of collaborative filtering prediction will become relatively dense, which can reduce the sparsity of the matrix and make collaborative filtering more accurate. In this way, we will try to improve the system performance through the integration of the two.

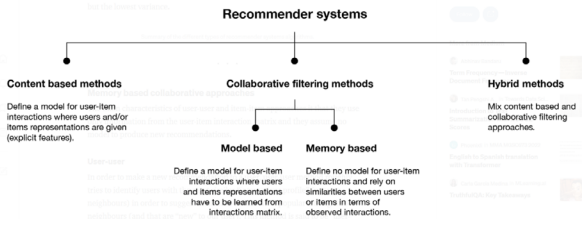


Fig. 5. Difference between User-User and Item-Item filtering

IV. COMPARISONS

The user-user method is based on the search of similar users in terms of interactions with items. As, in general, every user have only interacted with a few items, it makes the method pretty sensitive to any recorded interactions (high variance). On the other hand, as the final recommendation is only based on interactions recorded for users similar to our user of interest, we obtain more personalized results (low bias).

Conversely, the item-item method is based on the search of similar items in terms of user-item interactions. As, in general, a lot of users have interacted with an item, the neighborhood search is far less sensitive to single interactions (lower variance). As a counterpart, interactions coming from every kind of users (even users very different from our reference user) are then considered in the recommendation, making the method less personalized (more biased). Thus, this approach is less personalized than the user-user approach but more robust.

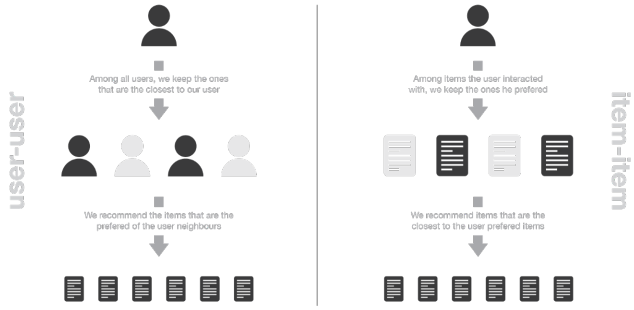


Fig. 5. Difference between User-User and Item-Item filtering

Collaborative Filtering vs. Content Filtering :

An Recommender System that suggests items to a user based on past interactions between users and items is called a Collaborative Filtering system.In these recommendation engines, a matrix of user-item interactions is created, in which every pair of users and items has a place.The space is either filled with the user's rating of that item or left blank.We will use this when we develop our models to perform matrix factorization or nearest neighbor classification.Collaborative filtering requires only the user id, item id, and rating fields.Collaborative filtering requires only the user id, item id, and rating fields.

While content filtering focuses exclusively on either the item or the user, it does not need to know about interactions between the two.Content filtering calculates similarity between items or users based on attributes of those items or users.As part of my book data, I will use book reviews and text analysis to determine which books are most similar to the books I like and, therefore, which books should be recommended (item-based).

V. CONCLUSION

We need to be able to analyze the performance of our recommender systems, just like any other machine learning algorithm, in order to select which algorithm is appropriate for our case. The two types of evaluation approaches for recommender systems are evaluations based on well-defined metrics and evaluations based mostly on human judgment and satisfaction estimation.

Explainability is another key point of the success of recommendation algorithms. Indeed, it has been proven that if users do not understand why they had been recommended as specific item, they tend to loose confidence into the recommender system.

If our recommender system is based on a model that generates numeric values such as ratings forecasts or matching probabilities, we may analyze the accuracy of these outputs using a standard error assessment metric like mean square error (MSE). In this situation, the model is only trained on a portion of the available interactions before being tested on the rest.

In this project, after evaluating several methods for recommendation. It was observed that combining the content based and collaborative methods have significantly improved the recommendations. Since there are no numerical recommendations in this project, instead we are recommending titles to users, it gets difficult to evaluate the model comprehensively. However, the recommendations generated looks relevant as per user history and content of the similar books. It is also observed that we get different recommendations for different user, even though the input title is the same.

VI. REFERENCES

[1] R. J. Vidmar. (1992, Aug.). On the use of atmospheric plasmas as electromagnetic reflectors. *IEEE Trans. Plasma Sci.* [Online]. *21(3),* pp. 876–880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar

[2] P. Li and Z. Hong, "Book recommendation algorithm based on the interest and type factor for university," Journal of Zhejiang university of technology, vol. 47, no. 4, pp. 425–429, 2019.

[3] X. Weng and Z. Wang, "Research progress of collaborative filtering recommendation algorithm," Computer Engineering and Applications, vol. 54, no. 1, pp. 25–31, 2018.

[4] J. He, "Collaborative filtering personalized recommendation algorithm based on popularity division combined with average preference weight," Computer Science, vol. 45, no. 6, pp. 25–33, 2018.

[5] M. Li, "Research on book combination recommendation system model based on tag and association rule mining," Computer Application Research, vol. 31, no. 8, pp. 2390–2393, 2014.