

Book Recommender System Using Collaborative Filtering

By-Devika R



Table of Contents

1. Introduction

2. Recommendation System Overview

- 2.1 Definition and Purpose

- 2.2 Types of Recommendation Systems

3. Book Recommender System with Collaborative Filtering

4. Data Collection and Preprocessing

- 4.1 Datasets

- 4.2 Preprocessing Steps

5. Model Implementation

- 5.1 Creating the User-Book Interaction Matrix

- 5.2 Building the Collaborative Filtering Model

6. Deployment

7. Conclusion

# 1. Introduction

This document details the implementation of a Book Recommender System using collaborative filtering. The system is designed to suggest books to users based on their reading history and preferences.

# 2. Recommendation System Overview

## 2.1 Definition and Purpose

A recommendation system is a subclass of information filtering systems that seeks to predict the "rating" or "preference" a user would give to an item. The main purpose of a recommendation system is to enhance the user's experience by suggesting items that are most relevant to their tastes and preferences.

## 2.2 Types of Recommendation Systems

There are primarily two types of recommendation systems:

### Collaborative Filtering

Collaborative filtering methods build a model from a user's past behavior (items previously purchased or rated) as well as similar decisions made by other users. There are two types of collaborative filtering:

- User-Based Collaborative Filtering: Recommends items based on similarity between users.

- Item-Based Collaborative Filtering: Recommends items based on similarity between items.

### Content-Based Filtering

Content-based filtering methods use item features to recommend other items similar to what the user likes, based on their previous actions or explicit feedback. For instance, if a user likes a book, the system will recommend other books that have similar attributes (e.g., genre, author).

# 3. Book Recommender System with Collaborative Filtering

Our Book Recommender System employs collaborative filtering to provide personalized book recommendations. The system uses user ratings to find similarities between users and books, enabling it to suggest books that users with similar tastes have liked.

# 4. Data Collection and Preprocessing

## 4.1 Datasets

The dataset used for this project is obtained from [https://www.kaggle.com/datasets/ra4u12/bookrecommendation].

The project uses three datasets:

-BX- Books: Contains information about the books.

- BX-Book-Ratings: Contains user ratings for different books.

- BX-Users: Contains information about the users.

## 4.2 Preprocessing Steps

1. Filtering Users: Select users who have rated more than 200 books.

2. Merging Datasets: Merge the ratings dataset with the books dataset to include book information.

3. Filtering Books: Filter books that have been rated more than 50 times to ensure a good amount of information.

# 5. Model Implementation

## 5.1 Creating the User-Book Interaction Matrix

Create a pivot table where rows represent book titles and columns represent user IDs, with the values being the ratings.

```python

book\_pivot = final\_rating.pivot\_table(columns='user\_id', index='title', values='rating')

```

Convert the pivot table to a sparse matrix to optimize for memory usage.

```python

from scipy.sparse import csr\_matrix

book\_sparse = csr\_matrix(book\_pivot)

```

## 5.2 Building the Collaborative Filtering Model

Use the Nearest Neighbors algorithm to find similar books.

Generate book recommendations based on a given book.

```python

def recommend\_book(book\_name):

book\_id = np.where(book\_pivot.index == book\_name)[0][0]

distances, suggestions = model.kneighbors(book\_pivot.iloc[book\_id, :].values.reshape(1, -1), n\_neighbors=6)

for i in range(len(suggestions)):

books = book\_pivot.index[suggestions[i]]

for j in books:

print(j)

```

# 6. Deployment

Deploy the recommendation system using a web framework like Flask. Example deployment steps:

1. Save the trained model and necessary data structures using pickle.

```python

import pickle

pickle.dump(model, open('model.pkl', 'wb'))

pickle.dump(book\_pivot, open('book\_pivot.pkl', 'wb'))

```

2. Create a Flask application to serve the recommendations.

```python

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

@app.route('/recommend', methods=['GET'])

def recommend():

book\_name = request.args.get('book\_name')

recommendations = recommend\_book(book\_name)

return jsonify(recommendations)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

```

# 7. Conclusion

The Book Recommender System effectively provides personalized book recommendations using collaborative filtering. The system leverages user rating data to identify similar books and users, offering tailored suggestions.