

Project Report BDA  
 Semester Project

GROUP MEMBERS:

|  |  |  |
| --- | --- | --- |
| MOHSIN KHAN | 373403 | BSCS-11-B |
| ALLAH RAKHA | 379284 | BSCS-11-B |
| ADNAN HASSNAIN | 391302 | BSCS-11-B |

Contents

[Semester Project 1](#_Toc167395176)

[Introduction 3](#_Toc167395177)

[2. Dataset Overview 3](#_Toc167395178)

[Key Attributes in the Dataset 3](#_Toc167395179)

[3. System Architecture 4](#_Toc167395180)

[3.1 Backend - Hybrid\_Rec\_System.ipynb 4](#_Toc167395181)

[Data Preprocessing 4](#_Toc167395182)

[Model Training 4](#_Toc167395183)

[Hybrid Approach 4](#_Toc167395184)

[Generating Recommendations 5](#_Toc167395185)

[3.2 Frontend - app.py 5](#_Toc167395186)

[Key Functions 5](#_Toc167395187)

[4. Evaluation 5](#_Toc167395188)

[Second Technique: Association Rule Mining Using FP-Growth 6](#_Toc167395189)

[Technique Two: 6](#_Toc167395190)

[1. Overview of Association Rule Mining 6](#_Toc167395191)

[5. Implementation Details 6](#_Toc167395192)

[5.1 Data Preparation 6](#_Toc167395193)

[5.2 Applying the FP-Growth Algorithm 6](#_Toc167395194)

[5.3 Interpreting the Results 6](#_Toc167395195)

[6. Advantages and Use Cases 6](#_Toc167395196)

[7. Conclusion 7](#_Toc167395197)

# Introduction

The Hybrid Book Recommendation System is designed to provide personalized book recommendations by leveraging a combination of collaborative filtering and content-based filtering techniques. This system utilizes a dataset containing information about books, such as ratings, reviews, and other metadata, to generate tailored suggestions for users.

**Objective:**

To design a comprehensive use case where students are required to find appropriate dataset(s) and apply multiple (2 or more) techniques studied in the class on the selected dataset(s) for the proposed use case.

* The more comprehensive the use case requiring maximum techniques will get more points.
* The use case can be implemented in Python but some part of the project should be implemented in either Hadoop or Spark.

# 2. Dataset Overview

The dataset used for this recommendation system is sourced from Kaggle, titled "Best Book Ever Data for 2021" by Shashwat Work. The dataset includes various attributes about books, which are essential for building a robust recommendation system.

# Key Attributes in the Dataset

* **Title**: The title of the book.
* **Author**: The author of the book.
* **Genre**: The genre or genres the book belongs to.
* **Publisher**: The publishing house of the book.
* **User Rating**: Ratings given by users.
* **Reviews**: Number of reviews a book has received.
* **Price**: Price of the book.
* **Year**: Year of publication.

# 3. System Architecture

The Hybrid Book Recommendation System is implemented using two main components:

* **Backend (Hybrid\_Rec\_System.ipynb)**: This notebook handles data processing, model training, and recommendation logic.
* **Frontend (app.py)**: This script provides a user interface for interacting with the recommendation system.

# 3.1 Backend - Hybrid\_Rec\_System.ipynb

The backend notebook is responsible for the following tasks:

# Data Preprocessing

* **Loading Data**: Reading the dataset into a pandas DataFrame.
* **Cleaning Data**: Handling missing values, normalizing text, and preparing data for analysis.
* **Feature Engineering**: Creating additional features that can be useful for recommendations, such as combining genres or extracting features from text.

# Model Training

The recommendation system employs two types of models:

* **Collaborative Filtering**: Utilizes user-item interactions (ratings) to find similarities between users or items. The Alternating Least Squares (ALS) algorithm is commonly used here.
* **Content-Based Filtering**: Recommends items based on the similarity of item features. Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) and cosine similarity are applied to text data from book titles, authors, and genres.

# Hybrid Approach

Combines the strengths of both collaborative and content-based filtering to provide more accurate and diverse recommendations. This can be achieved through various methods such as weighted averaging of scores from both models or switching between models based on the availability of data.

# Generating Recommendations

* **User-Specific Recommendations**: Generating a list of recommended books for a specific user based on their past interactions and preferences.
* **Item-Specific Recommendations**: Finding similar books to a given book using content-based filtering.

# 3.2 Frontend - app.py

The frontend script creates a web application using Flask, enabling users to interact with the recommendation system. Key functionalities include:

* **User Interface**: Simple and intuitive UI for users to input their preferences and receive recommendations.
* **API Endpoints**: Backend integration to fetch recommendations and display them to users.
* **Interactive Elements**: Features like search bars, dropdowns, and buttons to enhance user experience.

# Key Functions

* **Home Route**: Displays the main page where users can input their preferences.
* **Recommendation Route**: Handles user input, queries the backend for recommendations, and displays the results.

# 4. Evaluation

The effectiveness of the recommendation system can be evaluated using metrics such as:

* **Precision@k**: Measures the proportion of relevant items among the top-k recommendations.
* **Recall@k**: Measures the proportion of relevant items successfully recommended out of all relevant items.
* **RMSE (Root Mean Squared Error)**: Evaluates the prediction accuracy of the collaborative filtering model.

### Second Technique: Association Rule Mining Using FP-Growth

In this section, we explore an alternative method for book recommendation: Association Rule Mining using the FP-Growth algorithm. This technique is particularly useful for identifying patterns and associations between items in large datasets. By analyzing the co-occurrences of books read by users, we can uncover meaningful relationships and generate recommendations based on frequent itemset.

# Technique Two:

# 1. Overview of Association Rule Mining

Association Rule Mining is a technique used to find interesting relationships between variables in large datasets. In the context of book recommendations, it helps in discovering patterns of books that are frequently read together by users. The FP-Growth (Frequent Pattern Growth) algorithm is an efficient method for mining frequent itemsets without the need for candidate generation, making it suitable for large datasets.

# 5. Implementation Details

### 5.1 Data Preparation

The dataset needs to be preprocessed to fit the requirements of the FP-Growth algorithm. Specifically, we need to convert the list of books read by each user into an array format.

### 5.2 Applying the FP-Growth Algorithm

The FP-Growth algorithm is then applied to the preprocessed data to mine frequent itemsets and generate association rules. These rules help in understanding the relationships between different books and can be used to recommend books based on the books a user has already read.

### 5.3 Interpreting the Results

The frequent itemsets represent groups of books that are often read together, while the association rules indicate the likelihood of a user reading a certain book given that they have read other books. These insights can be leveraged to make data-driven recommendations.

# 6. Advantages and Use Cases

The FP-Growth algorithm and association rule mining provide several advantages:

• Scalability: Efficiently handles large datasets.

• Interpretability: Results are easy to understand and interpret.

• Actionable Insights: Provides concrete rules that can be directly used for recommendations.

This technique is particularly useful for generating recommendations based on historical co-occurrence patterns, making it a valuable component of a hybrid recommendation system.

By integrating association rule mining with other recommendation techniques, we can enhance the diversity and accuracy of recommendations, offering users a more comprehensive and satisfying

# 7. Conclusion

The Hybrid Book Recommendation System leverages both collaborative and content-based filtering techniques to provide personalized book recommendations. By combining these approaches, the system can offer more accurate and diverse suggestions, enhancing the user experience. The integration of a Flask-based web application allows users to interact with the system seamlessly, making it a practical solution for book recommendation needs.