NovelNest

Book Recommender System

Report Submission for Flipkart Grid 5.0

Team Name: ChangeMakers

Team Members:

- Mayuri Gund
- Divyashree Chavhan

Institute Name: Shrimati Kashibai Navale College Of Engineering

Date: 20/08/23
Location: Pune

Acknowledgment

I would like to take a moment to express my heartfelt appreciation to my project partner Divyashree Chavhan . Working on this project as a team of two has been a rewarding and enriching experience.Her dedication and expertise have been instrumental in bringing this project to fruition.

Divyashree has been an exceptional collaborator throughout this journey. Her dedication, expertise, and unwavering support have been instrumental in the successful completion of this project. We have shared ideas, overcome challenges, and worked together seamlessly to bring our product recommendation system to life.

I would also like to thank our users whose feedback and engagement have been invaluable in refining and enhancing our recommendation algorithms. Your participation has been crucial to the system's continuous improvement.

I am also thankful for the trust and autonomy we had in managing our project, which allowed us to make decisions effectively and implement our ideas with creativity.
Additionally, we acknowledge the support and resources provided by our organization, which enabled us to undertake this project and realize our vision.

Finally, I want to express my heartfelt gratitude to our mentors and advisors for their guidance and insights that have shaped the direction of this recommendation system.
This project represents the collective efforts of many, and I am thankful for the collaborative spirit and dedication of all involved."

Abstract

In our day-to-day life, every one of us happens to use e-commerce websites to shop for many items. This leads to an increasing diversity of consumer demand, turning into a challenge for a retail store to provide the right products accordingly to customer preferences. These e-commerce websites use different types of recommendation systems to provide a positive shopping experience to the user.

Recommender systems are a tool to cope with this challenge, through product recommendation it is possible to fulfill customers' needs and expectations, helping maintain loyal customers while attracting new customers. In this project, we have decided to work on a book recommendation system for the simplicity of the project. We have used the ratings given to a book(product) by other users to make recommendations to the current user.

We have created a matrix to represent the ratings given to the product by the different users. Next, we created the user profile and item profile and provide recommendations depending on the similarity and their past interactions. Then we apply collaborative, content-based algorithms which is a part of machine learning. To evaluate the performance of the model, RMSE was used which gives information on the closeness of recommendations being generated.

Introduction

A product recommendation is a filtering system that seeks to predict and show the items that a user would like to purchase. It may not be entirely accurate, but if it shows what a user like then it is doing its job right. Recommendation engines basically are data filtering tools that make use of algorithms and data to recommend the most relevant items to a particular user. In simple terms, they are nothing but an automated form of a “shop counter guy”.

Need of recommendation system

- In the immortal words of Steve Jobs "*A lot of times, people don’t know what they want until you show it to them.*” Customers may love your movie, your product, your job opening but they may not know it exists. The job of the recommender system is to open the customer/user up to completely new products and possibilities, which they would not think to directly search for themselves.

- With the growing amount of information on the internet and with a significant rise in the number of users, it is becoming important for companies to search, map and provide them with the relevant chunk of information according to their preferences and tastes.

Types of Recommendation System

There are majorly three important types of recommendation systems:

- **Collaborative filtering**
- **Content-Based Filtering.**
- **Hybrid Recommendation Systems**

1.Collaborative Based Filtering

- This filtering method is usually based on collecting and analyzing information on user's behaviors, their activities or preferences and predicting what they will like based on the similarity with other users
- This filtering method is usually based on collecting and analyzing information on user's behaviors, their activities or preferences and predicting what they will like based on the similarity with other users
- A key advantage of the collaborative filtering approach is that it does not rely on machine analyzable content and thus it is capable of accurately recommending complex items such as movies without requiring an "understanding" of the item itself.
- Collaborative filtering is based on the assumption that people who agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past.
- For example, if a person A likes item 1, 2, 3 and B like 2,3,4 then they have similarinterests and A should like item 4 and B should like item 1

Types of Recommendation System

2.Content Based Filtering

- These filtering methods are based on the description of an item and a profile of the user’s preferred choices

- In a content-based recommendation system, keywords are used to describe the items; besides, a user profile is built to state the type of item this user likes. In other words, the algorithms try to recommend products which are similar to the ones that a user has liked in the past

- The idea of content-based filtering is that if a user like an item then he/she will also like a ‘similar’ item.

- For example, when we are recommending the same kind of item like a movie or song recommendation. This approach has its roots in information retrieval and information filtering research

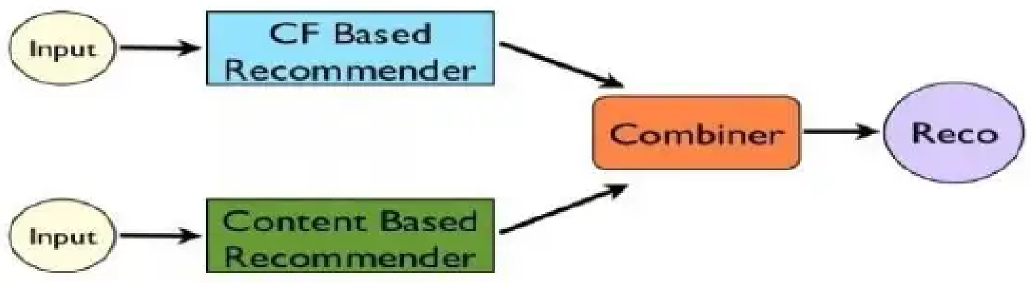
- A major issue with content-based filtering is whether the system is able to learn user preferences from users actions about one content source and replicate them across other different content types

- When the system is limited to recommending the content of the same type as the user is already using, the value from the recommendation system is significantly less when other content types from other services can be recommended. For example, recommending news articles based on the browsing of news is useful, but wouldn’t it be much more useful when music, videos from different services can be recommended based on the news browsing.

Types of Recommendation System

3.Hybrid Recommendation systems

- Hybrid approaches can be implemented by making content-based and collaborative-based predictions separately and then combining them
- Further, by adding content-based capabilities to a collaborative-based approach and viceversa; or by unifying the approaches into one model
- Several studies focused on comparing the performance of the hybrid with the pure collaborative and content-based methods and demonstrate that hybrid methods can provide more accurate recommendations than pure approaches
- Such methods can be used to overcome the common problems in recommendation systems such as cold start and the data paucity problem.
- Netflix is a good example of the use of hybrid recommender systems. The website makesrecommendations by comparing the watching and searching habits of similar users (i.e.,collaborative filtering) as well as by offering movies that share characteristics with filmsthat a user has rated highly (content-based filtering)



Approach

1 Data Collection and Preparation:

- User profiles, product categories, and interaction histories are defined and collected.
- User preferences, interaction types (e.g., viewed, purchased, rated), and product attributes are gathered.

2.Collaborative Filtering (CF):

Collaborative Filtering is a user-based recommendation technique that leverages the historical interactions between users and items. The CF approach consists of the following steps:

2.1 User-Item Interaction Matrix and Normalization:

- Construct a matrix with rows representing books and columns representing users.
- Populate the matrix with user ratings for books, filling unrated items with zeros.
- Calculate the average rating given by each user.
- Subtract the user's average rating from each of their individual ratings to obtain normalized ratings.

2.2 Similarity Calculation:

- Calculate the cosine similarity between books based on their ratings across users.
- The similarity score indicates how similar two books are in terms of user preferences.

2.3 User-Based Recommendation:

- For a given book that a user has liked, identify the row in the matrix representing that book.
- Recommend the top-N similar books to the user who liked the input book.

Approach

3.Popularity-Based Filtering (PBF):

Popularity-Based Filtering focuses on recommending items that are widely popular among users. The PBF approach involves the following steps:

3.1 Rating and Popularity Calculation:

- Calculate the average rating and the total number of ratings for each book.
- Create a popularity threshold to filter out less popular books**.**

3.2 Filtering and Recommendation:

- Filter out books with low average ratings and ratings below the popularity threshold.
- Generate a list of popular books that meet the filtering criteria.

.

Strengths

- **Personalization:** The system provides personalized recommendations based on both collaborative filtering and popularity-based filtering. This ensures that users receive recommendations tailored to their preferences and behaviors.

- **Diversity:** By combining collaborative and popularity-based approaches, the system offers a diverse set of recommendations. Collaborative filtering captures user preferences, while popularity-based filtering introduces variety by suggesting popular items.

- **User Engagement:** Recommending popular books can enhance user engagement, as these books are more likely to resonate with a wider audience. This can lead to higher user satisfaction and interaction.

- **Simplicity:** The popularity-based filtering approach is relatively simple to implement. It requires minimal computation and can quickly generate recommendations based on popularity thresholds.

- **Cold Start Handling:** Popularity-based recommendations are useful for new users or items that lack sufficient interaction data. This addresses the "cold start" problem, where collaborative filtering struggles due to limited data.

Limitations

- **Cold Start for New Users and Items:** Collaborative filtering requires a history of user interactions to generate meaningful recommendations. New users and items face the "cold start" problem, as there might be insufficient data for accurate predictions.
- **Sparsity:** Collaborative filtering suffers from sparsity when the user-item interaction matrix has a lot of empty entries (unrated items). This can lead to challenges in accurately calculating similarities.
- **Popularity Bias:** Popularity-based filtering tends to recommend only well-known items, potentially overshadowing niche or lesser-known gems that users might prefer.
- **Lack of Personalization in Popularity-Based Filtering:** Popularity-based filtering doesn't take into account individual user preferences and behaviors. Recommendations might not align with a user's unique tastes.
- **Data Scalability:** Both approaches might face scalability issues with large datasets. Collaborative filtering requires computing similarities for all item pairs, and popularity-based filtering involves calculating popularity metrics for each item.
- **Limited Context:** The current system mainly relies on book ratings and popularity metrics. It doesn't consider contextual factors such as the user's current interests, reading history, or specific genres.
- **Quality of Ratings:** Collaborative filtering assumes that all user ratings are equally reliable and valuable, which might not be the case. Some users might rate arbitrarily or unintentionally.

Appendix

```
import numpy as np
import pandas as pd

books = pd.read_csv('data/Books.csv')
users = pd.read_csv('data/Users.csv')
ratings = pd.read_csv('data/Ratings.csv')

books.head()

books.isnull().sum()

ratings_with_name = ratings.merge(books , on = 'ISBN')
ratings_with_name.isnull()

num_rating = ratings_with_name.groupby('Book-Title').count()['Book-
Rating'].reset_index()
num_rating.rename(columns={'Book-Rating' : 'num_Ratings'} , inplace=True)

avg_rating = ratings_with_name.groupby('Book-Title')['Book-
Rating'].mean().reset_index()
avg_rating.rename(columns={'Book-Rating' : 'avg_Ratings'} , inplace=True)
avg_rating

##Popular Books

popular_df = num_rating.merge(avg_rating, on = 'Book-Title')
popular_df

filtered_popular_df = popular_df[popular_df['num_Ratings'] >= 250]
sorted_filtered_df = filtered_popular_df.sort_values('avg_Ratings', ascending=False)
top_50_books = sorted_filtered_df.head(50)

print(top_50_books)

top_50_books = top_50_books.merge(books , on = 'Book-
Title').drop_duplicates('Book-Title')[['Book-Title' , 'Book-Author' , 'Image-URL-M' ,
'num_Ratings' , 'avg_Ratings']]
```

```
#Collaborative Filtering

x = ratings_with_name.groupby('User-ID').count()['Book-Rating'] > 200

padhe_likhe_users = x[x].index

filtered_rating = ratings_with_name[ratings_with_name['User-
ID'].isin(padhe_likhe_users)]

y = filtered_rating.groupby('Book-Title').count()['Book-Rating'] >= 50
famous_books = y[y].index

final_ratings = filtered_rating[filtered_rating['Book-Title'].isin(famous_books)]

pt = final_ratings.pivot_table(index = 'Book-Title' ,columns = 'User-ID' , values = 'Book-
Rating')

pt.fillna(0 , inplace = True)

from sklearn.metrics.pairwise import cosine_similarity

similarity_score = cosine_similarity(pt)

similarity_score.shape


def recommend(book_name):
   # index fetch
   index = np.where(pt.index==book_name)[0][0]
   similar_items = sorted(list(enumerate(similarity_score[index])),key=lambda
x:x[1],reverse=True)[0:5]

   data = []
   for i in similar_items:
      item = []
      temp_df = books[books['Book-Title'] == pt.index[i[0]]]
      item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-Title'].values))
      item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-Author'].values))
      item.extend(list(temp_df.drop_duplicates('Book-Title')['Image-URL-M'].values))

      data.append(item)

   return data
```

recommend('1984')

Generated Recommendations from Same author or Same genre

```
[['1984',
  'George Orwell',
  'http://images.amazon.com/images/P/0451524934.01.MZZZZZZZ.jpg'],
 ['Animal Farm',
  'George Orwell',
  'http://images.amazon.com/images/P/0451526341.01.MZZZZZZZ.jpg'],
 ["The Handmaid's Tale",
  'Margaret Atwood',
  'http://images.amazon.com/images/P/0449212602.01.MZZZZZZZ.jpg'],
 ['Brave New World',
  'Aldous Huxley',
  'http://images.amazon.com/images/P/0060809833.01.MZZZZZZZ.jpg'],
 ['The Vampire Lestat (Vampire Chronicles, Book II)',
  'ANNE RICE',
  'http://images.amazon.com/images/P/0345313860.01.MZZZZZZZ.jpg']]
```

Thank You !!