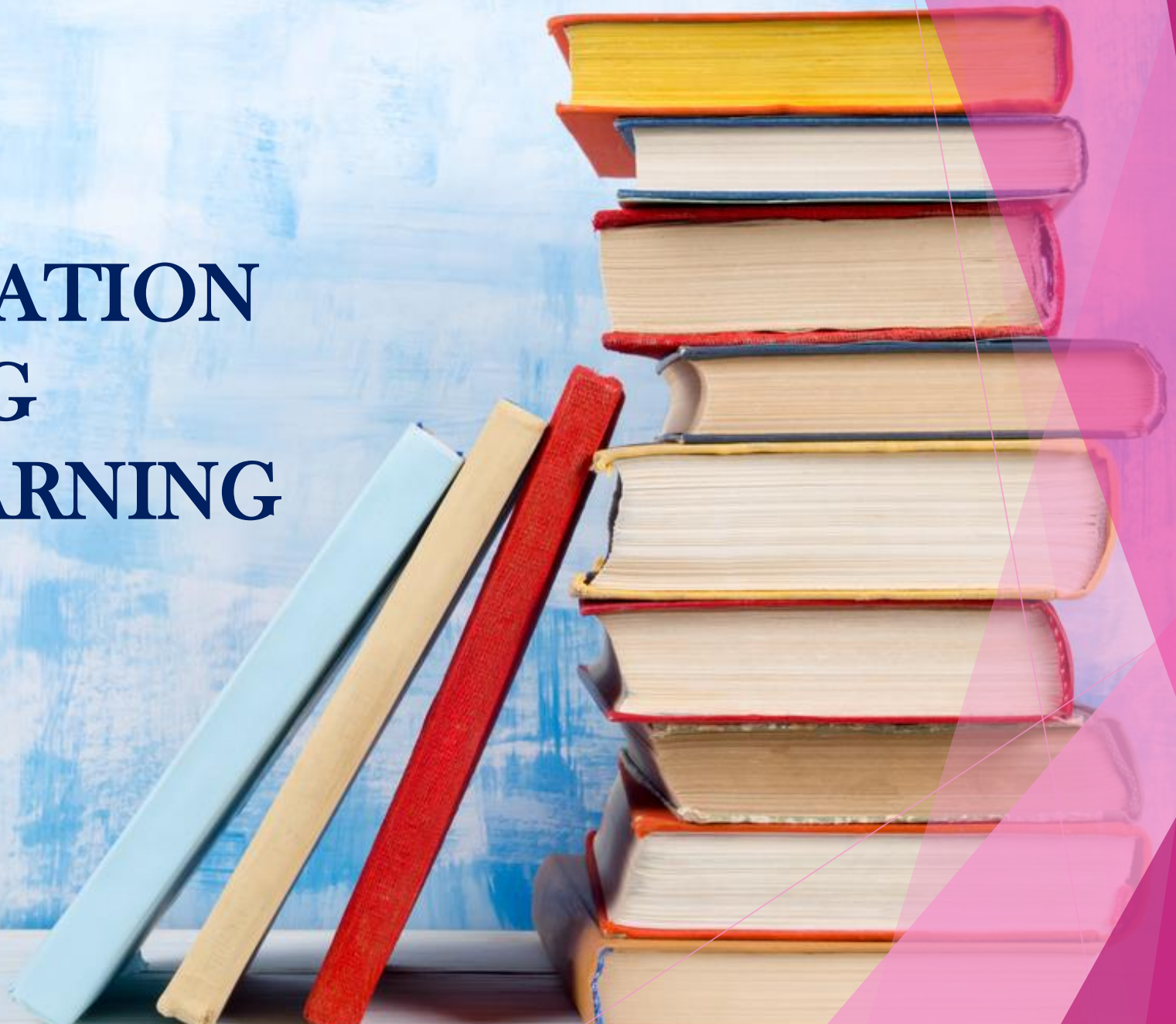
BOOK RECOMMENDATION SYSTEM USING MACHINE LEARNING



GROUP - 3 TEAM MEMBERS NAMES:

DANDU SUMATHI

GANTA PRAVALLIKA

KALAGA SRIVATSAV

M ARAVIND

SOUMYA HAVANAGI

TABLE OF CONTENT:

INTRODUCTION:

A book recommendation system suggests books to users based on their interests. It uses data like user ratings, reviews, and book details to provide personalized recommendations. By analyzing this data, the system can predict which books a user might enjoy, enhancing their reading experience.

Why Book Recommendation Systems?

User Engagement: Encourages users to discover and read more books by suggesting titles that align with their interests.

Personalization: Tailors suggestions to individual user preferences, making the user experience more enjoyable and unique.

Increased Sales/Usage: Increases the likelihood of users purchasing or borrowing more books, thereby boosting sales or library usage.

Datasets:

Users_dataset:

- User-ID (unique for each user)
- Location (contains city, state and country separated by commas)

Books_dataset:

- ISBN (unique for each book)
- Book-Title
- Book-Author
- Year-Of-Publication

Ratings_dataset:

- User-ID
- ISBN
- Book ratings

PROJECT GOAL:

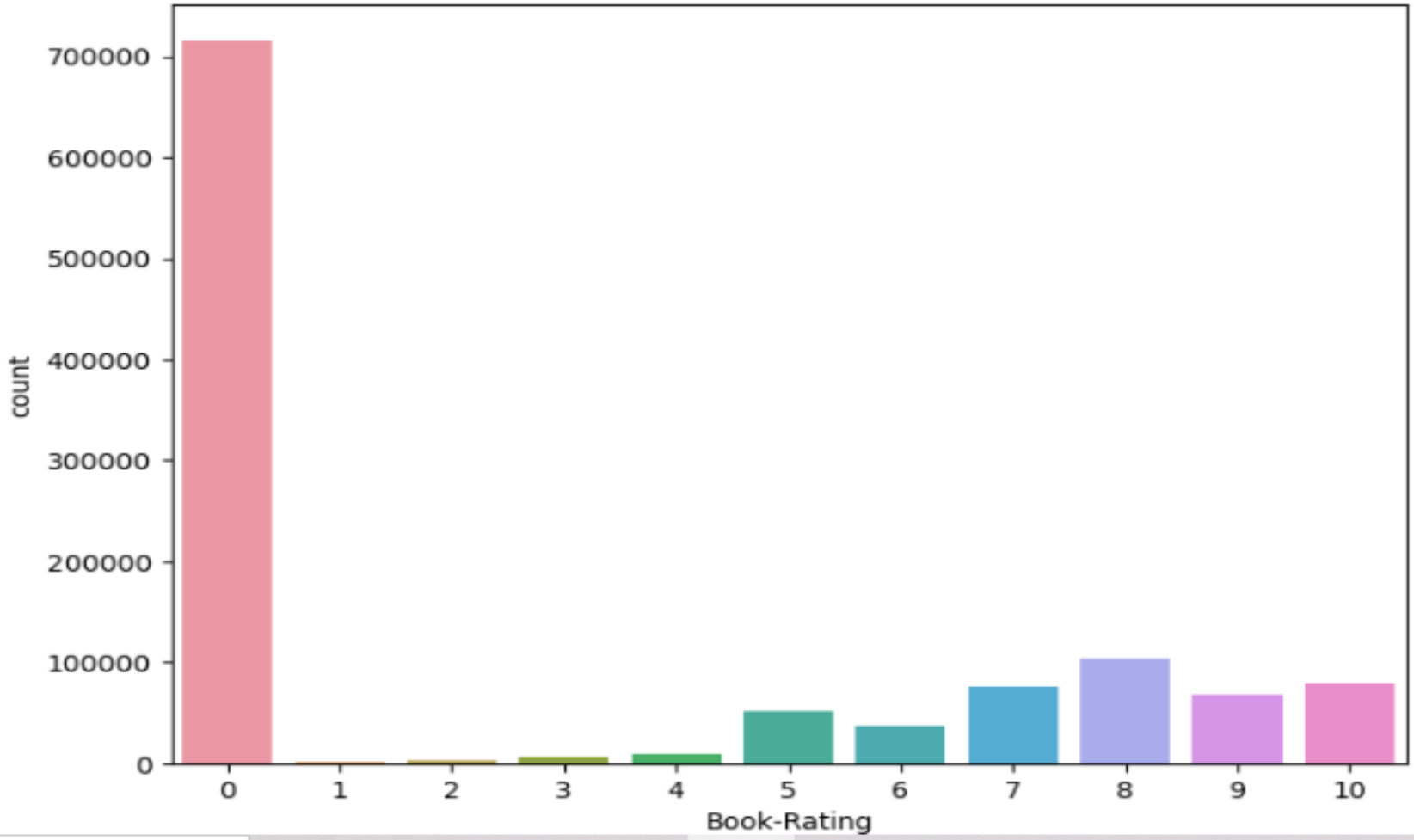
Explore and compare different approaches to recommending the most relevant books to users based on their interactions(ratings) with other books in the past or based on their similar users interactions with books

EDA
(Exploratory Data Analysis)

Visualising Rating Counts
Using Seaborn Library for ploting a countplot

```
]: # countplot on Book Rating
   plt.figure(figsize=(8,6))
   sns.countplot(x="Book-Rating", data=ratings)
```

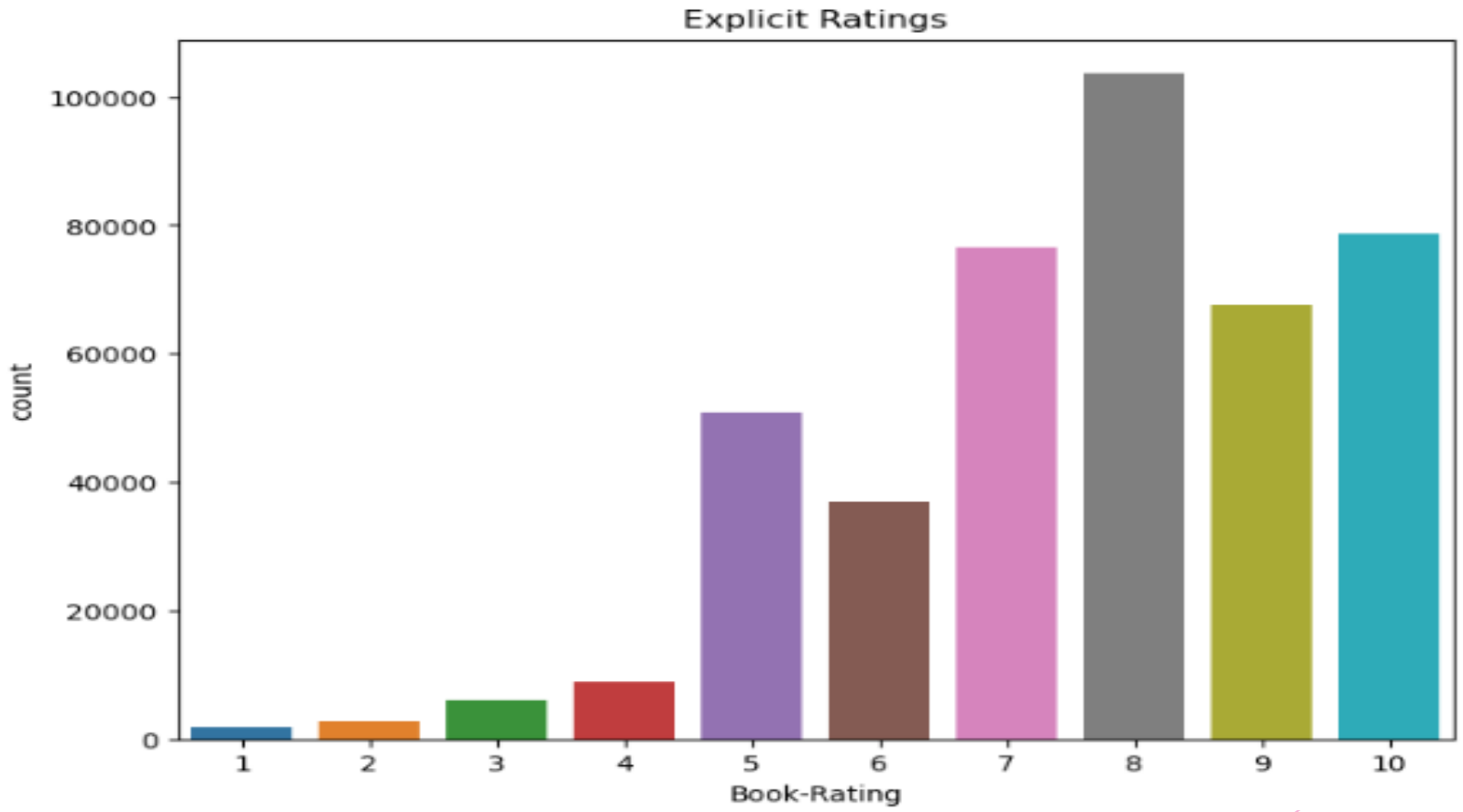
```
]: <Axes: xlabel='Book-Rating', ylabel='count'>
```



Visualising Explicit Rating Counts
Using Seaborn Library for ploting a countplot

```
plt.figure(figsize=(8,6))
data = ratings[ratings['Book-Rating'] != 0]
sns.countplot(x="Book-Rating", data=data)
plt.title("Explicit Ratings")
```

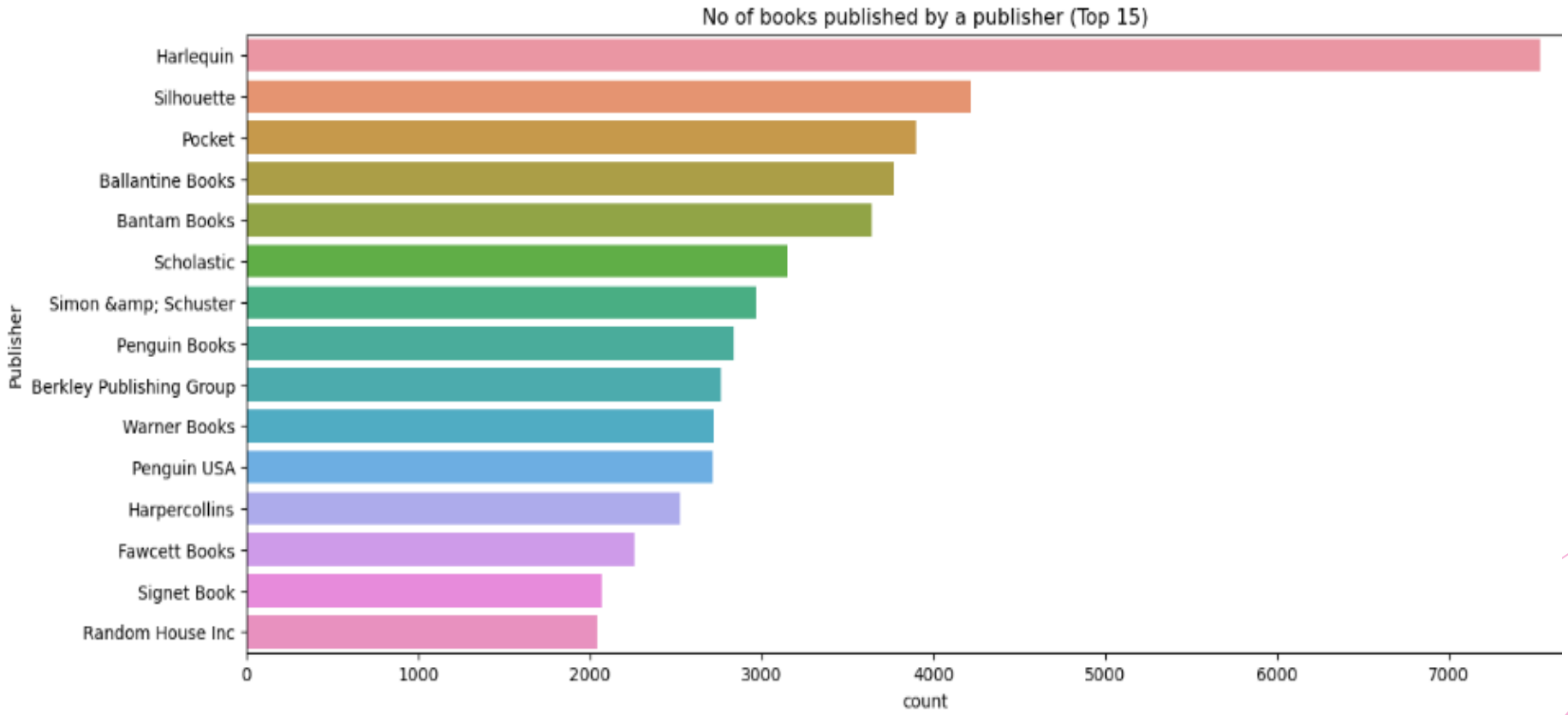
```
Text(0.5, 1.0, 'Explicit Ratings')
```



Visualizing (top 15) number of books published by publisher

```
plt.figure(figsize=(15,6))
sns.countplot(y="Publisher", data=books,order=books['Publisher'].value_counts().index[0:15])
plt.title("No of books published by a publisher (Top 15)")
```

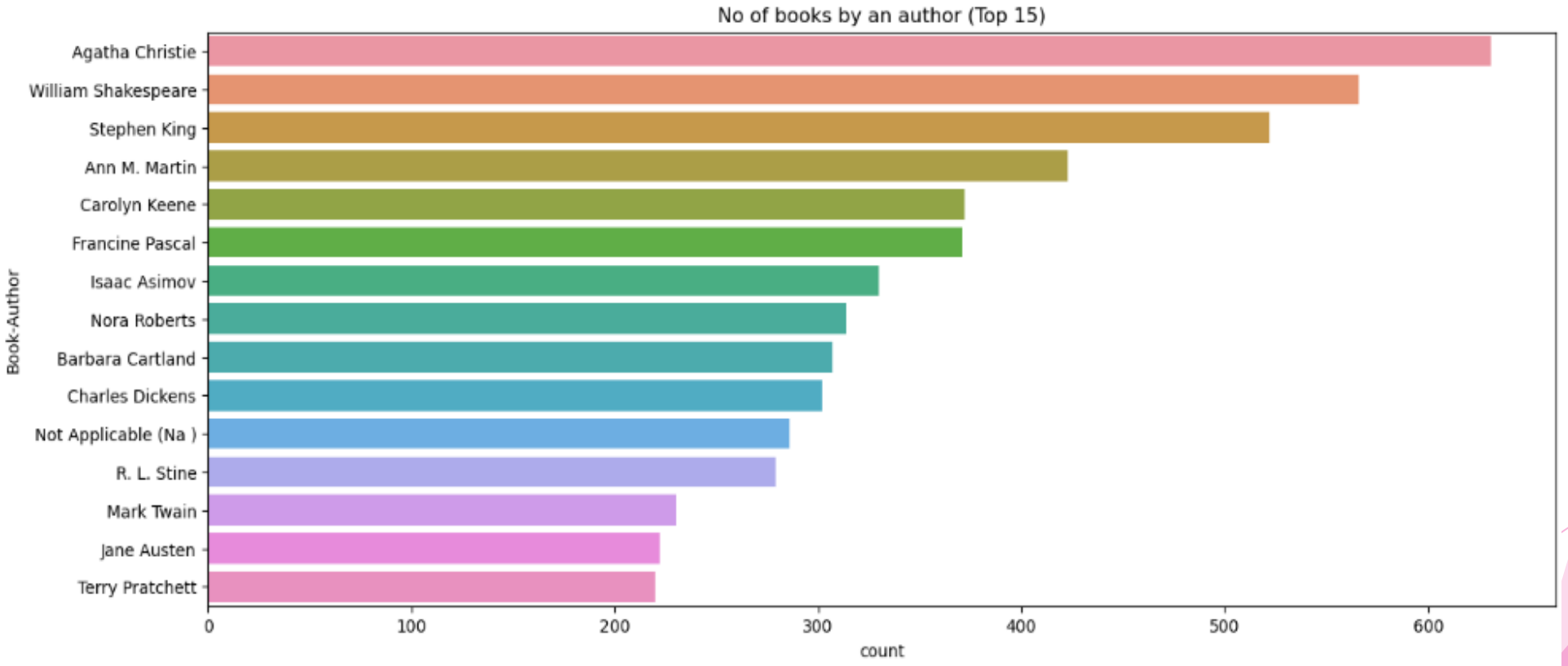
```
Text(0.5, 1.0, 'No of books published by a publisher (Top 15)')
```



Visualizing (top 15) number of books wrote by author

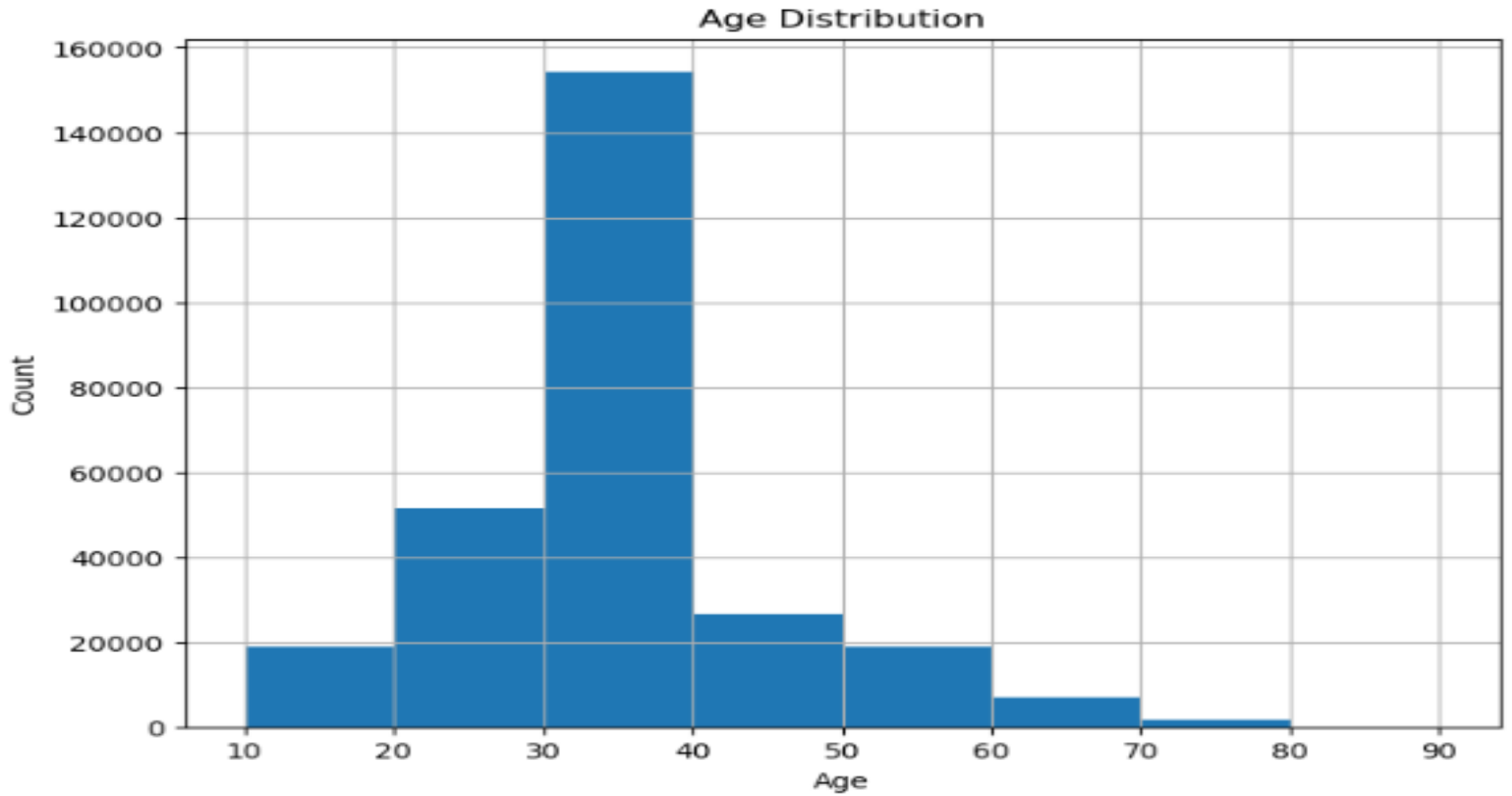
```
plt.figure(figsize=(15,6))
sns.countplot(y="Book-Author", data=books,order=books['Book-Author'].value_counts().index[0:15])
plt.title("No of books by an author (Top 15)")
```

```
Text(0.5, 1.0, 'No of books by an author (Top 15)')
```



Visualizing the age distribution of the users

```
plt.figure(figsize=(8,6))
users.Age.hist(bins=[10*i for i in range(1, 10)])
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



MODEL BUILDING

Model Building:

Popularity Based Recommender System:

In this model, we recommended the books according to their popularity score i.e. based on the average rating the book received. We considered only those books whose ratings are more than 250 and received the top 50 books accordingly.

- **Considering those books whose book rating is more than 250**

```
#filtering books with num_rating>250 and then sorting the books and displaying top 50 books

popular_df = popular_df[popular_df['book_rating']>=250].sort_values('avg_rating',ascending=False).head(50)
popular_df
```

TOP 10 POPULAR BOOKS :

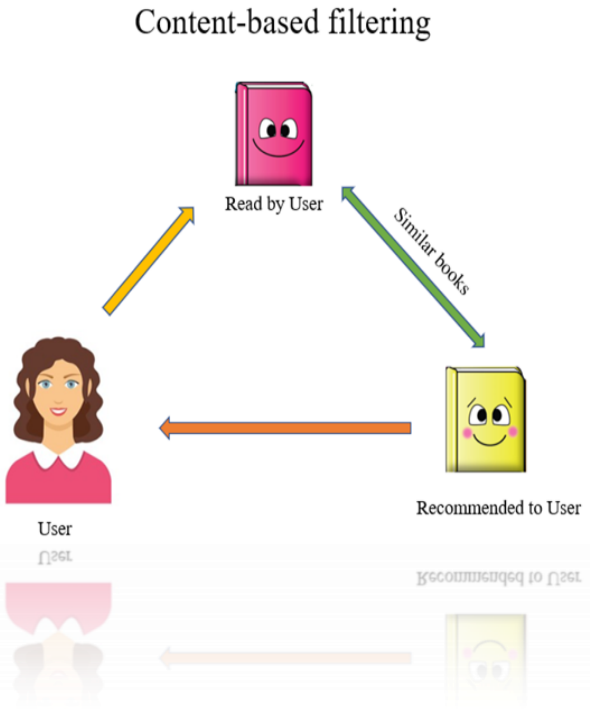
	Book-Title	Total-Ratings	Average Rating	score
1877	Harry Potter and the Chamber of Secrets Postcard Book	23	9.869565	9.450512
111	The Two Towers (The Lord of the Rings, Part 2)	136	9.330882	9.266768
3852	Dilbert: A Book of Postcards	13	9.923077	9.256352
1802	Calvin and Hobbes	24	9.583333	9.228080
4587	Postmarked Yesteryear: 30 Rare Holiday Postcards	11	10.000000	9.225896
2255	The Authoritative Calvin and Hobbes (Calvin and Hobbes)	20	9.600000	9.184573
1971	My Sister's Keeper : A Novel (Picoult, Jodi)	22	9.545455	9.170901
178	The Return of the King (The Lord of the Rings, Part 3)	103	9.213592	9.135318
3012	The Return of the King (The Lord of The Rings, Part 3)	16	9.625000	9.124492
1580	The Giving Tree	26	9.423077	9.116591

Different Filtering Techniques

Content-Based Filtering:

In Content-based recommender system, user provides data either explicitly (rating) or implicitly (By clicking on a link). The system captures this data and generates user profile for every user. By making use of user profile, recommendations are generate

In content-based filtering, recommendation is given by only watching single user's profile. System tries to recommend item similar to that item based on users past activity.

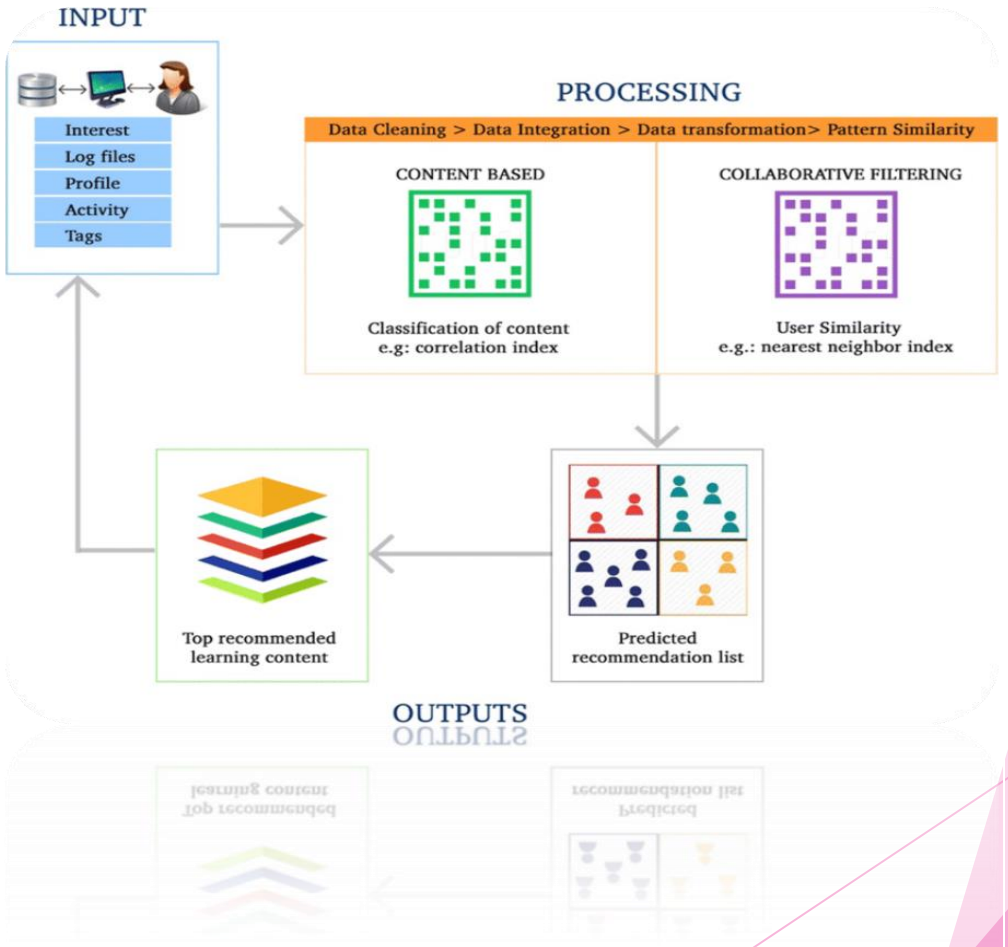


Collaborative Filtering:

Collaborative filtering is a technique that can filter out items that a user might like on the basis of reactions by similar users. It works by searching a large group of people and finding a smaller set of users with tastes similar to particular user

Hybrid Filtering:

A hybrid filtering approach is taken between context based filtering and collaborative filtering to implement the system. We can use content based filtering first and then pass those results to collaborative recommender (and vice-versa) or by integrating both the filter into one model to generate the results.



Collaborative Filtering Based Recommender System :

In this model, we used a Product-to-Product Based recommendation system. Here we recommended the books according to their similarity score. For that, we used a cosine-based similarity score.

```
from sklearn.metrics.pairwise import cosine_similarity
```

```
similarity_score = cosine_similarity(pt)
similarity_score
```

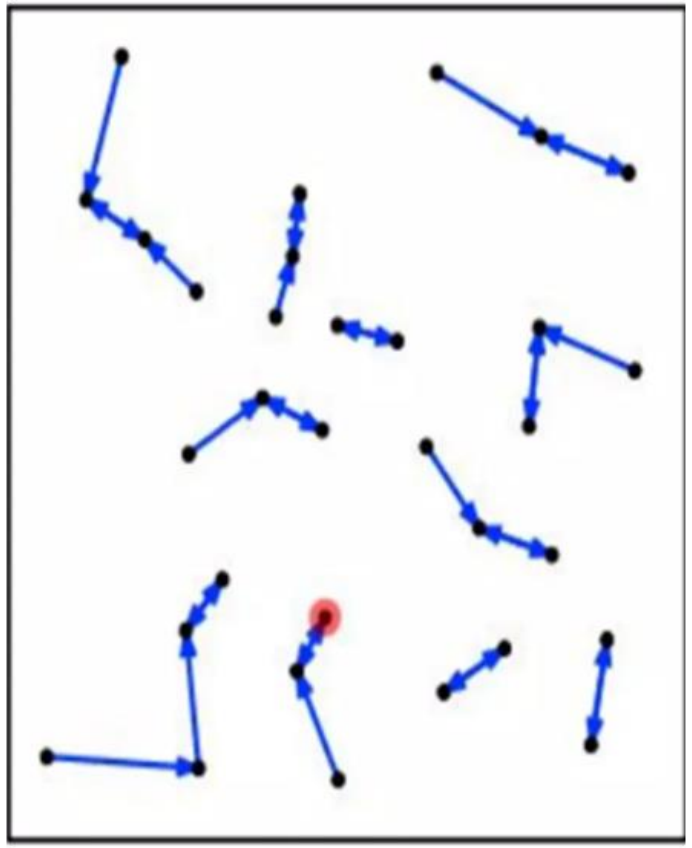
```
array([[1.        , 0.10255025, 0.01220856, ..., 0.12110367, 0.07347567,
        0.04316046],
       [0.10255025, 1.        , 0.2364573 , ..., 0.07446129, 0.16773875,
        0.14263397],
       [0.01220856, 0.2364573 , 1.        , ..., 0.04558758, 0.04938579,
        0.10796119],
```

K Nearest neighbors

The K-nearest neighbors(KNN) algorithm is a data classification method for estimating the likelihood that a data point will become a member of one group or another based on what group the data points nearest to it belong to.

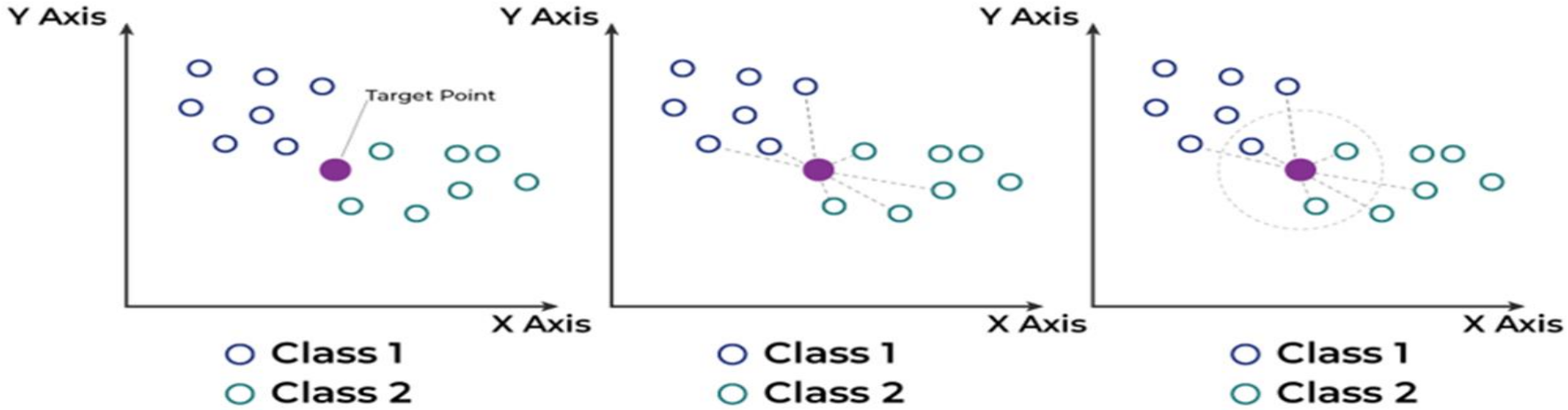
The principle behind nearest neighbor methods is to find a predefined number of samples closest in distance to the new point and predict labels from these.

The number of samples (K) can be a user-defined constant, or vary based on the local density of points(radius-based neighbor learning)



Working of KNN algorithm

The K-Nearest Neighbors (KNN) algorithm operates on the principle of similarity, where it predicts the label or value of a new data point by considering the labels or values of its K nearest neighbors in the training dataset.



```python
def recommend_book(book_name):
    book_id = np.where(book_pivot.index == book_name)[0][0]
    distance , suggestion = model.kneighbors(book_pivot.iloc[book_id,:].values.reshape(1,-1),n_neighbors=9)
    for i in range(len(suggestion)):
        books = book_pivot.index[suggestion[i]]
        for j in books:
            print(j)
```

```python
np.where(book_pivot.index == "Harry Potter and the Goblet of Fire (Book 4)")[0][0]
```

```
473
```

```python
book_name = 'Harry Potter and the Goblet of Fire (Book 4)'
recommend_book(book_name)
```

```
Harry Potter and the Goblet of Fire (Book 4)
Harry Potter and the Prisoner of Azkaban (Book 3)
Harry Potter and the Chamber of Secrets (Book 2)
Harry Potter and the Order of the Phoenix (Book 5)
Harry Potter and the Sorcerer's Stone (Book 1)
Special Delivery: A Novel
Once in a Lifetime
Star
Monster Blood (Goosebumps, No 3)
```

MODEL DEPLOYMENT

MODEL DEPLOYMENT

Model Deployment Using Streamlit:

We created the necessary pickle files to deploy our model on Streamlit.

```
import pickle
import os

os.makedirs("artifacts",exist_ok=True)
pickle.dump(model , open('artifacts/model.pkl','wb'))
pickle.dump(books_name, open('artifacts/books_name.pkl','wb'))
pickle.dump(df,open('artifacts/final_rating.pkl','wb'))
pickle.dump(book_pivot,open('artifacts/book_pivot.pkl','wb'))
```

Code using stream lit Library:

We used Spyder IDE for the deployment code.

Two Sections :

1.Similar books

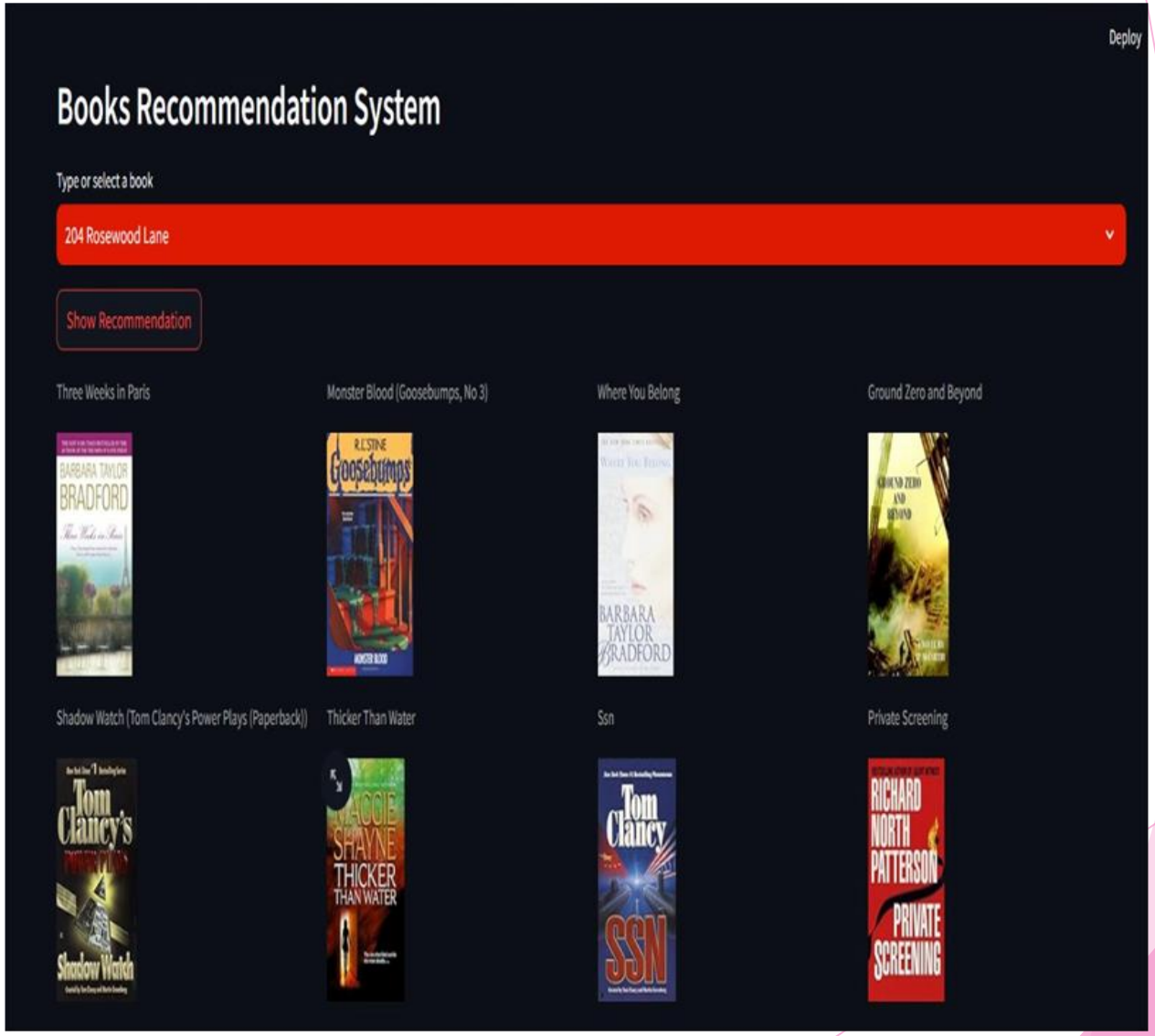
2.Book Recommender – Collaborative Filtering-Based Recommendation System

CODE:

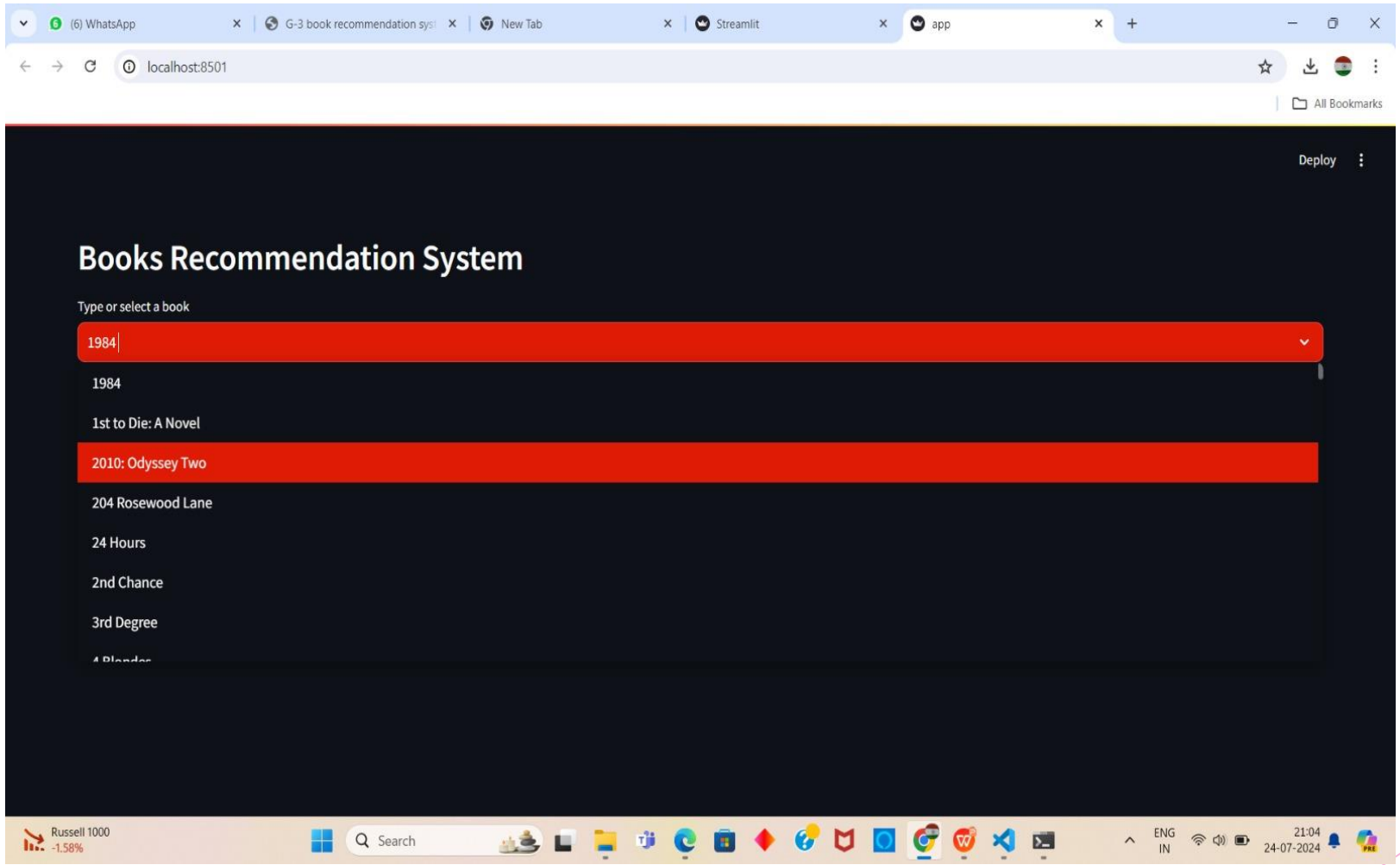
```python
import pickle
import streamlit as st
from utils import recommend_books


st.header("Books Recommendation System")

model = pickle.load(open('artifacts/model.pkl','rb'))
books_name = pickle.load(open('artifacts/books_name.pkl','rb'))
final_rating = pickle.load(open('artifacts/final_rating.pkl','rb'))
book_pivot = pickle.load(open('artifacts/book_pivot.pkl','rb'))


selected_book = st.selectbox("Type or select a book",books_name)

if st.button("Show Recommendation"):
    recommendation_books , poster_url = recommend_books(selected_book,model,books_name,final_rating,book_pivot)

    col1, col2 , col3, col4 = st.columns(4)
    col5, col6, col7, col8 = st.columns(4)

    with col1:
        st.caption(recommendation_books[1])
        st.image(poster_url[1])

    with col2:
        st.caption(recommendation_books[2])
        st.image(poster_url[2])

    with col3:
        st.caption(recommendation_books[3])
        st.image(poster_url[3])

    with col4:
        st.caption(recommendation_books[4])
        st.image(poster_url[4])

    with col5:
        st.caption(recommendation_books[5])
        st.image(poster_url[5])

    with col6:
        st.caption(recommendation_books[6])
        st.image(poster_url[6])

    with col7:
        st.caption(recommendation_books[7])
        st.image(poster_url[7])

    with col8:
        st.caption(recommendation_books[8])
        st.image(poster_url[8])
```

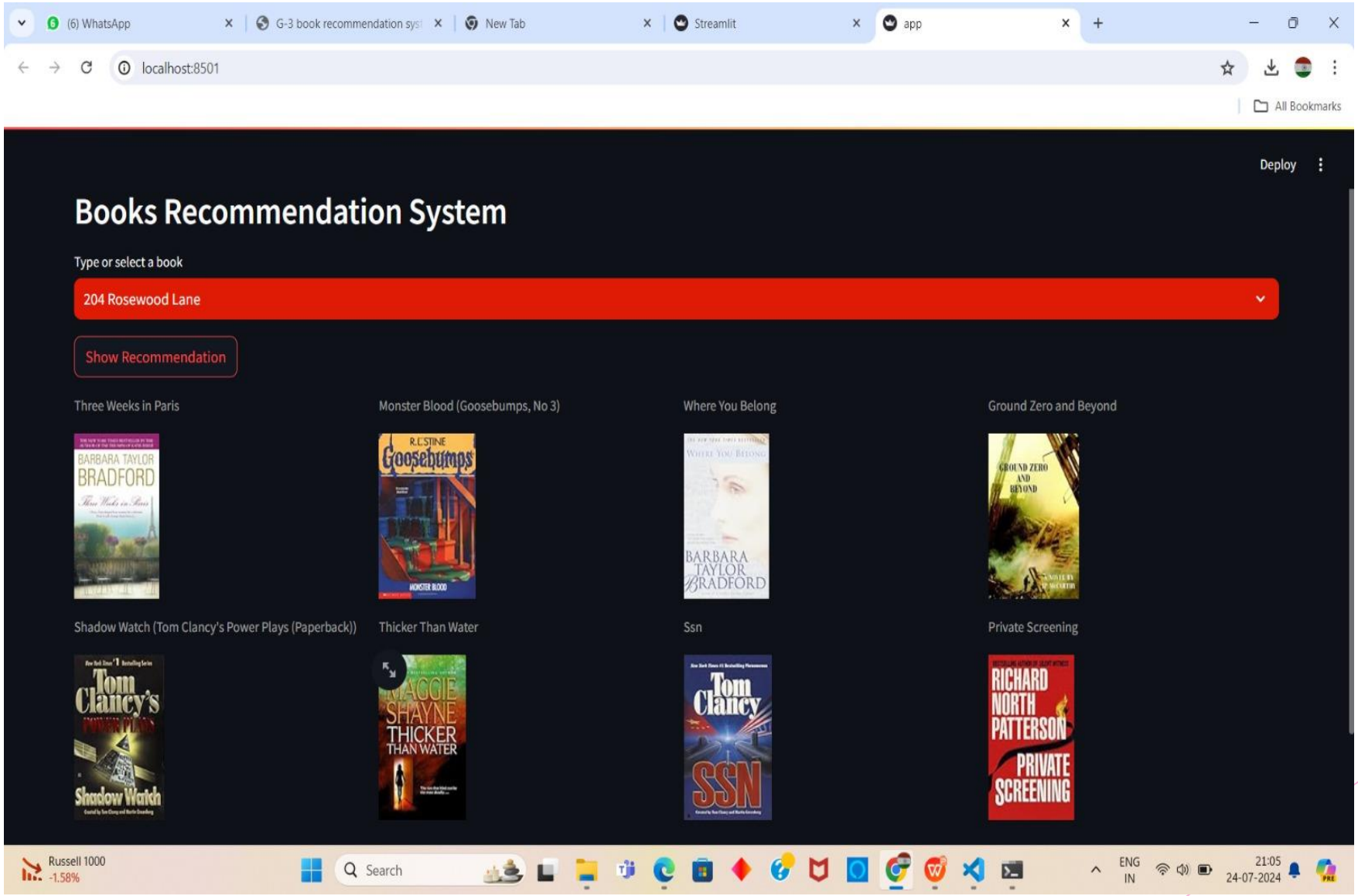
Top 8 similar books:

Books are Recommended according to the user input book name it will recommend the similar books according to the user book name



Book Recommender:




(6) WhatsApp
G-3 book recommendation syst
New Tab
Streamlit
app
localhost:8501
All Bookmarks
Deploy
Books Recommendation System
Type or select a book
204 Rosewood Lane
Show Recommendation
Three Weeks in Paris
Monster Blood (Goosebumps, No 3)
Where You Belong
Ground Zero and Beyond
Shadow Watch (Tom Clancy's Power Plays (Paperback))
Thicker Than Water
Ssn
Private Screening
Russell 1000
-1.58%
Search
ENG
IN
21:05
24-07-2024

CONCLUSION:

Book Recommendation Project successfully demonstrates the capability to enhance user experience by providing personalized book suggestions. By leveraging collaborative filtering and content-based algorithms, we created a robust system that analyzes user preferences and book features to recommend titles that align with individual tastes.Overall, this project lays a strong foundation for creating a dynamic and user-centric book recommendation system that can continuously evolve and improve. Thank you for your attention.