Institute of Advanced Computing and Software Development, Pune

Post Graduation Diploma in Big Data Analytics
Feb-2020

Book Recommendation System

By :
Gayatri Badgujar - 1308
Priyanka Kothawade - 1325

Project Guide
Mr. AKASHY TILEKAR

Contents

- Objectives
- Introduction
- Flow of execution
- Requirements
- Dataset Description
- Electronic design automation(EDA)
- Methods and Algorithm
- Result
- Conclusion

Introduction

- **Book Recommendation system** is used to recommend books to the users based on their interest and previous records. There are different ways to approach a recommendation system. They are basically divided into Content-based and Collaborative Filtering systems.

Objective

- ❑ Developing a machine learning recommendation system which recommends the books to the user based on their interests and previous records.
- ❑ Another objective is to build a recommendation system with good accuracy.

Flow of Execution

- ❑ Loading Books and Rating Datasets.
- ❑ Merging both dataset.
- ❑ Data Cleaning.
- ❑ Data Visualization.
- ❑ Recommendation Systems.

Requirements

❑ Data set : Books and Ratings Dataset

❑ Software : Jupyter Notebook, Google colab

❑ Programming language: Python and Machine learning

❑ Libraries: 1) Data visualization: Plotly, Matplotlib

2) Machine learning: Sklearn, pandas,

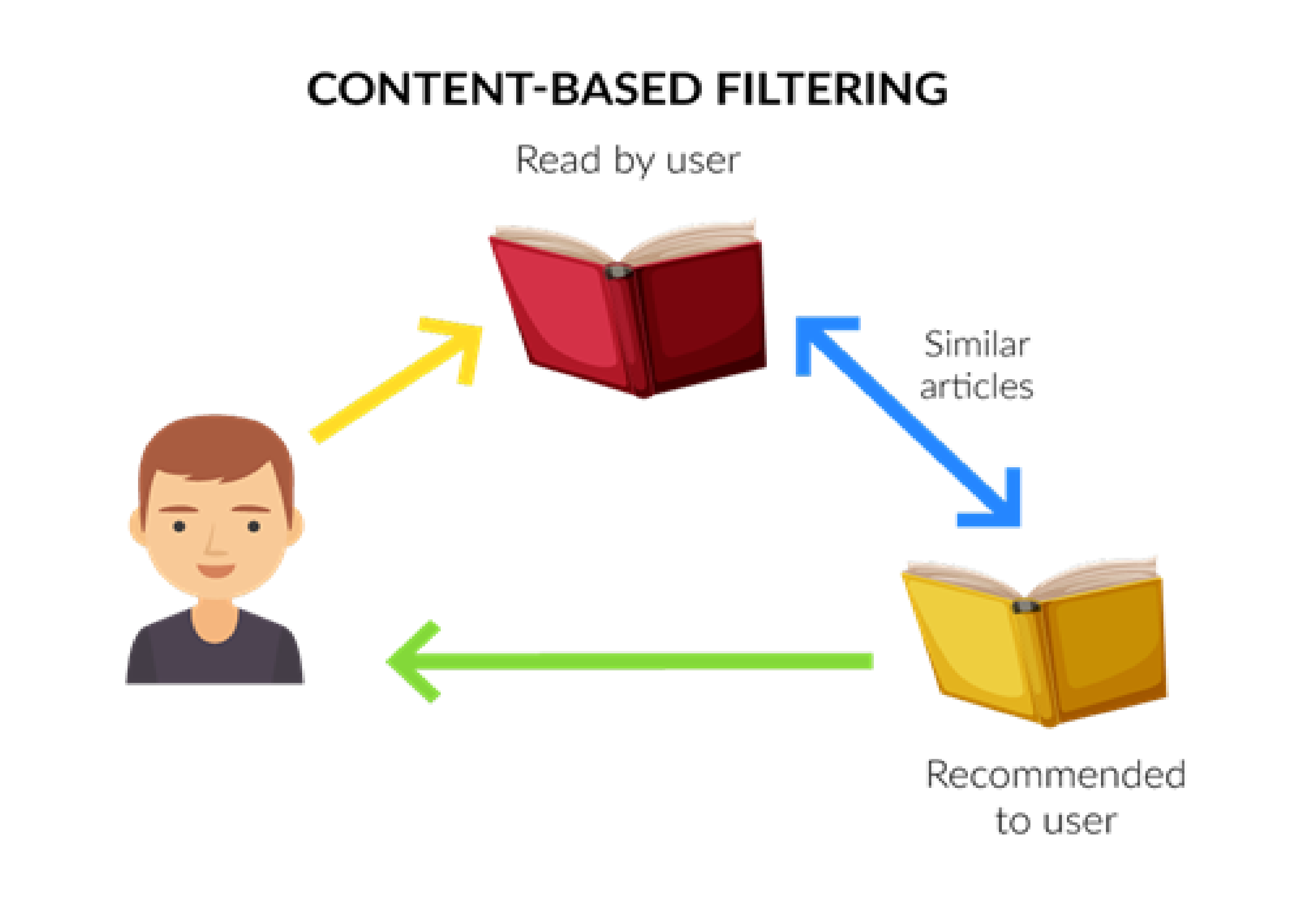
Surprise, NumPy

Dataset Description

- ❑ The BOOK Dataset consists of 10000 rows and 23 columns and The RATINGS dataset consists of 981756 rows and 3 columns. i.e. User-id Book-id and Ratings.
- ❑ We merged the books and ratings dataset for Data visualization and recommendation system. We merged these two datasets based on Book-id.
- ❑ After merging of two dataset the merged dataset consists of 23 columns.

Content based recommendation

- ❑ Content Based Systems recommend items similar to those a user has liked (browsed/purchased) in the past.
- ❑ Recommendations are based on the content of items rather on other user's opinion.



CONTENT-BASED FILTERING

Read by user

Similar articles

Recommended to user

Content Based Process - TF-IDF

❑ Each document is transformed into a normed TF-IDF vector, size N(Term Frequency/Inverse Document Frequency).

❑ The distance between any pair of vector is computed.

TF-IDF Weight = TF (t,d) * IDF(t,D)

Cosine Similarity

- ❑ Cosine similarity is a measure of similarity between two non zero vectors.
- ❑ By Vector representation we can see how closely related two sentence are based on what angles their respective vectors make.
- ❑ Cosine value ranges from -1 to 1.
- ❑ If two vectors make an angle 0 then the sentences are closely related to each other.
- ❑ If two vectors make an angle 90 then the sentences are almost unrelated.

Collaborative Filtering

- ❑ **Collaborative Filtering** Method finds a subset of users who have similar tastes and preferences to the target user and use this subset for offering recommendations.

- ❑ **Basic Assumptions:**

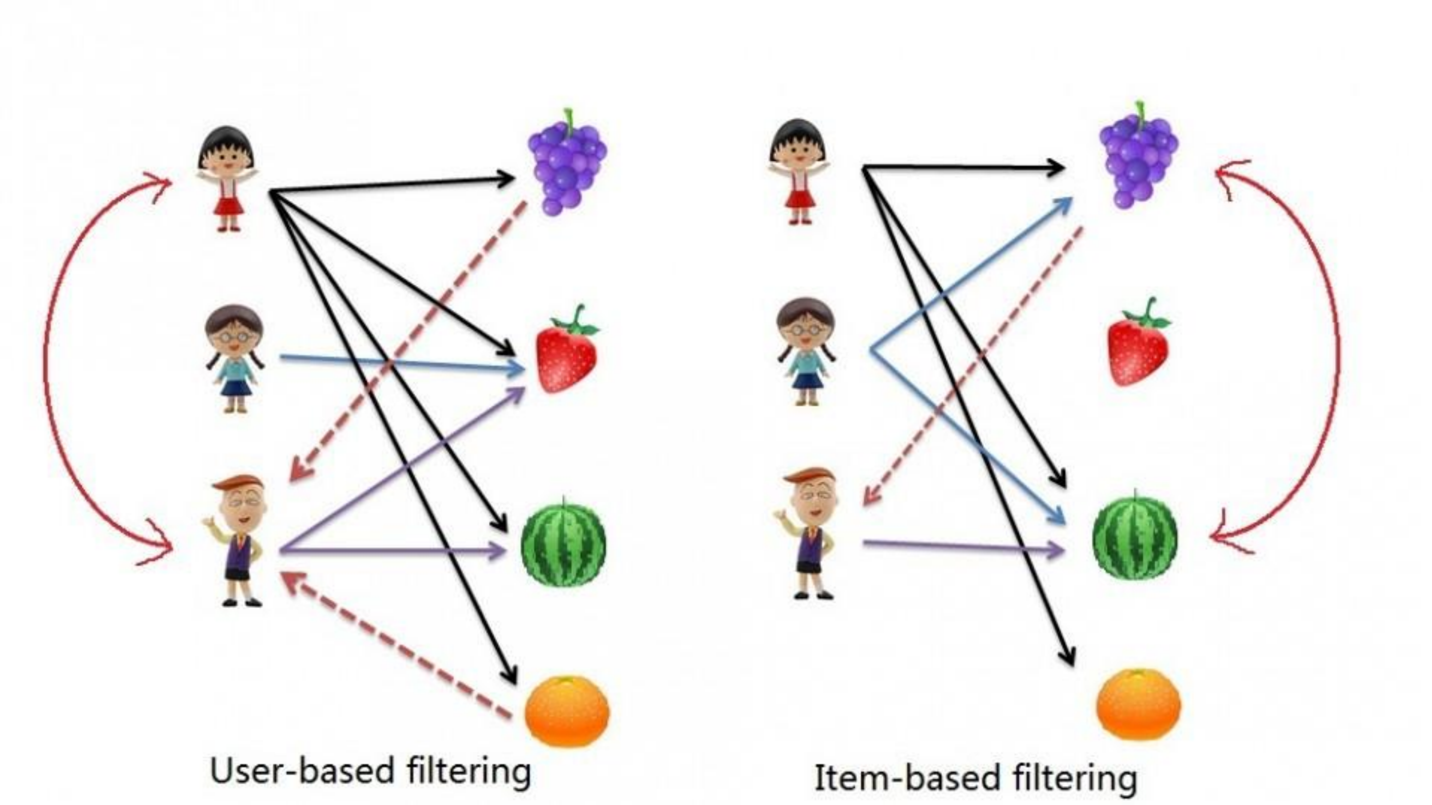
  -Users with similar interests have common preferences.

  -Sufficiently large number of user preferences are available.

- ❑ **Main Approaches:**

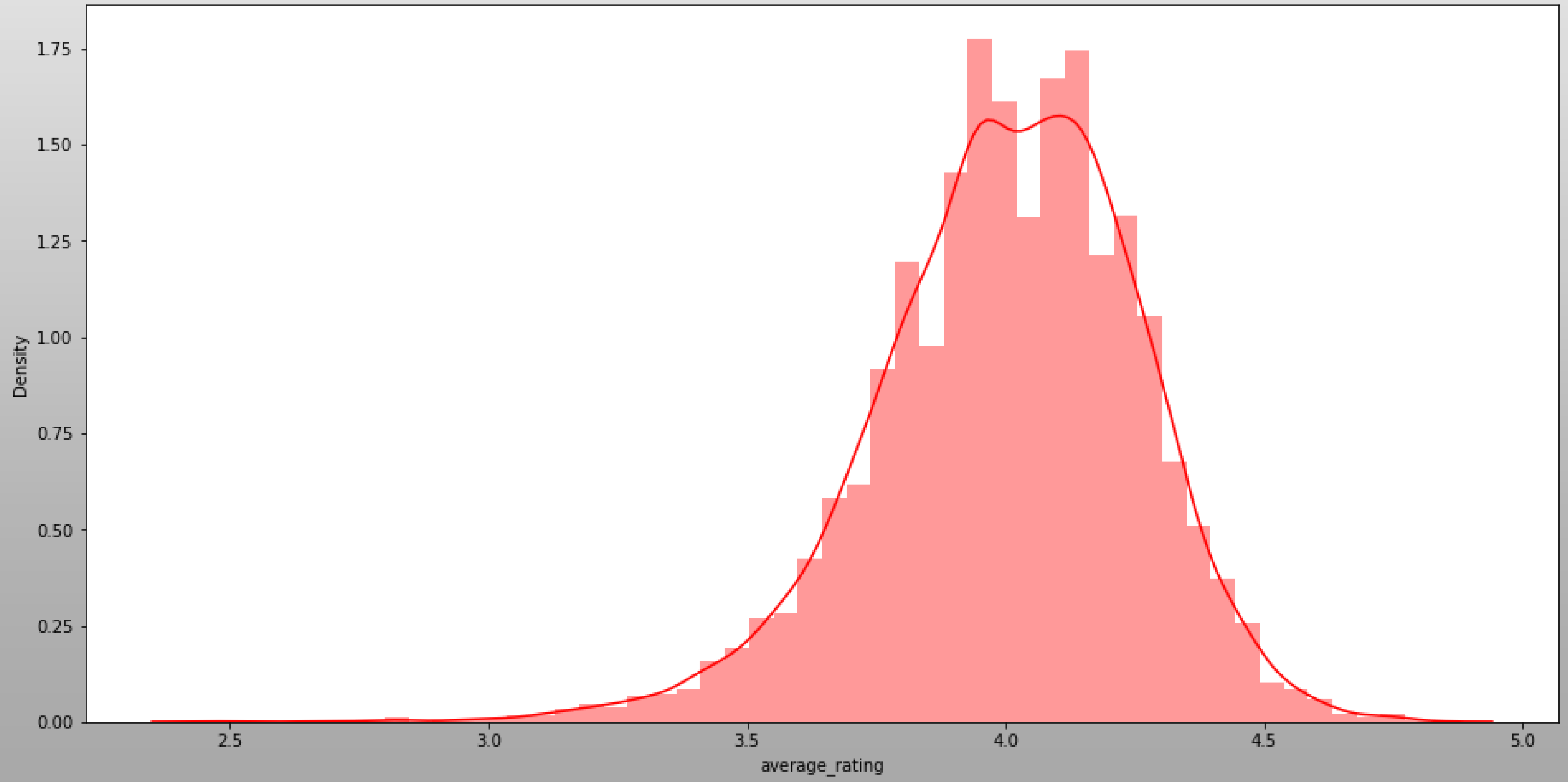
  -User Based

  -Item Based

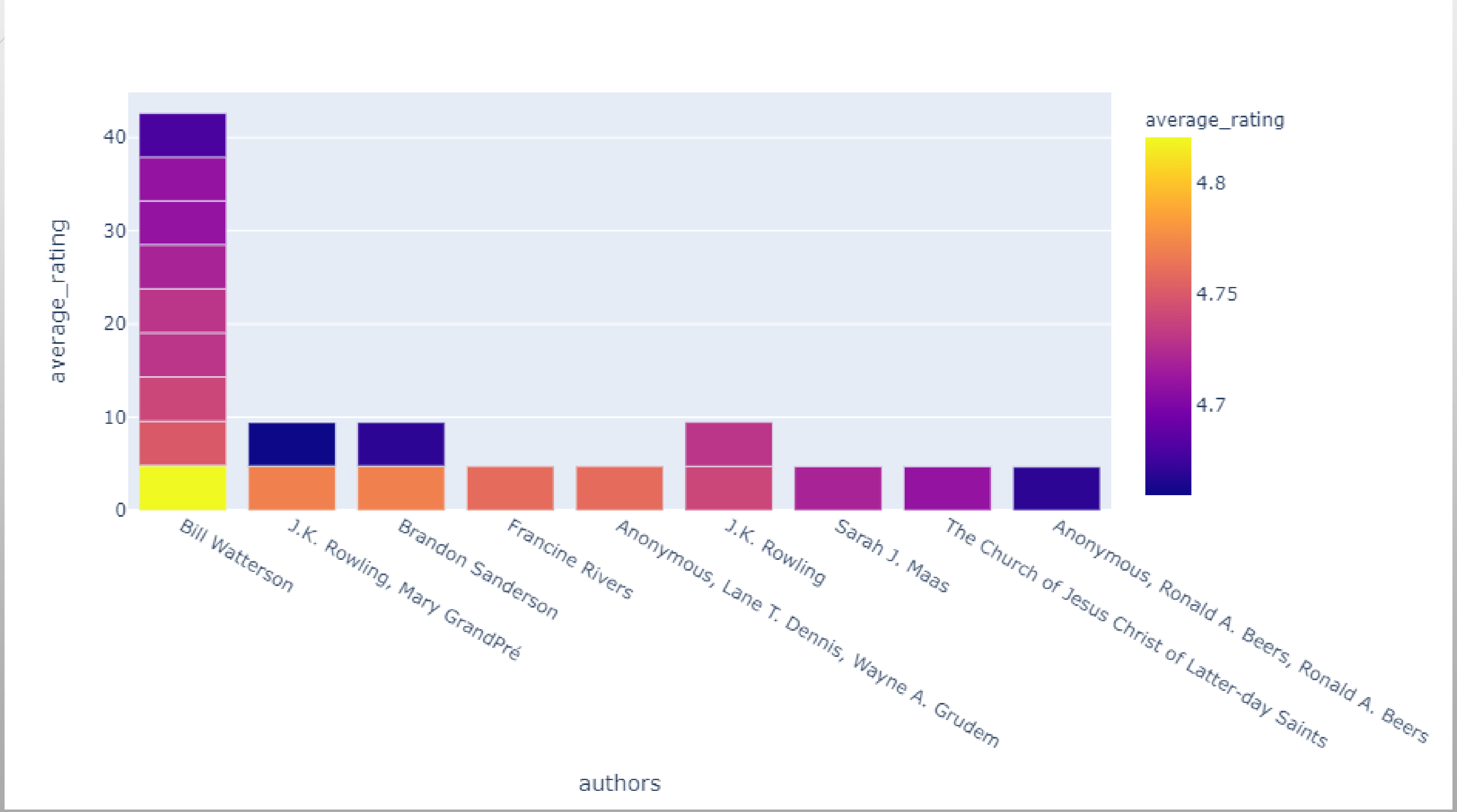

User-based filtering
Item-based filtering

EDA

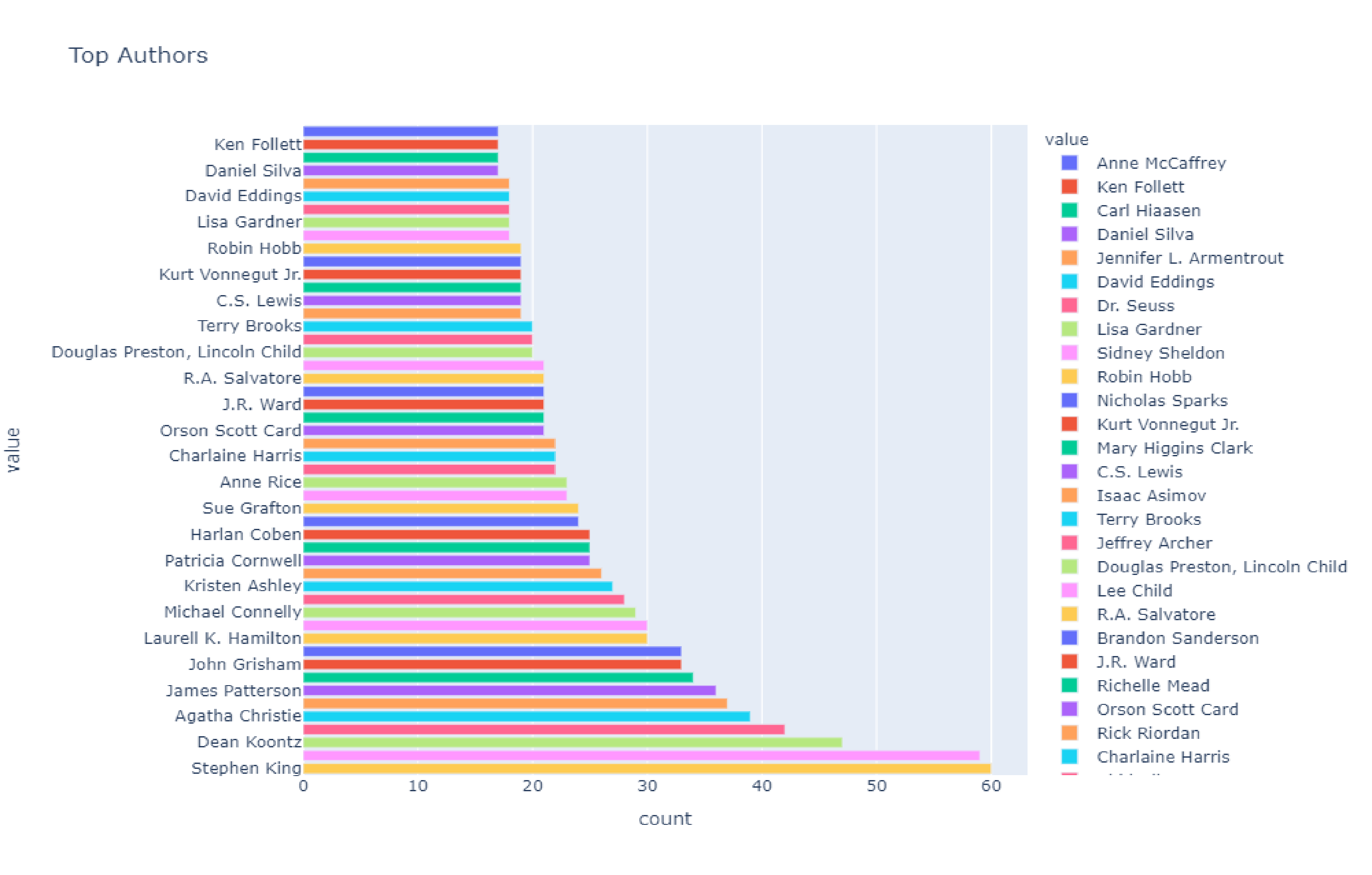
Most Common Rating Values



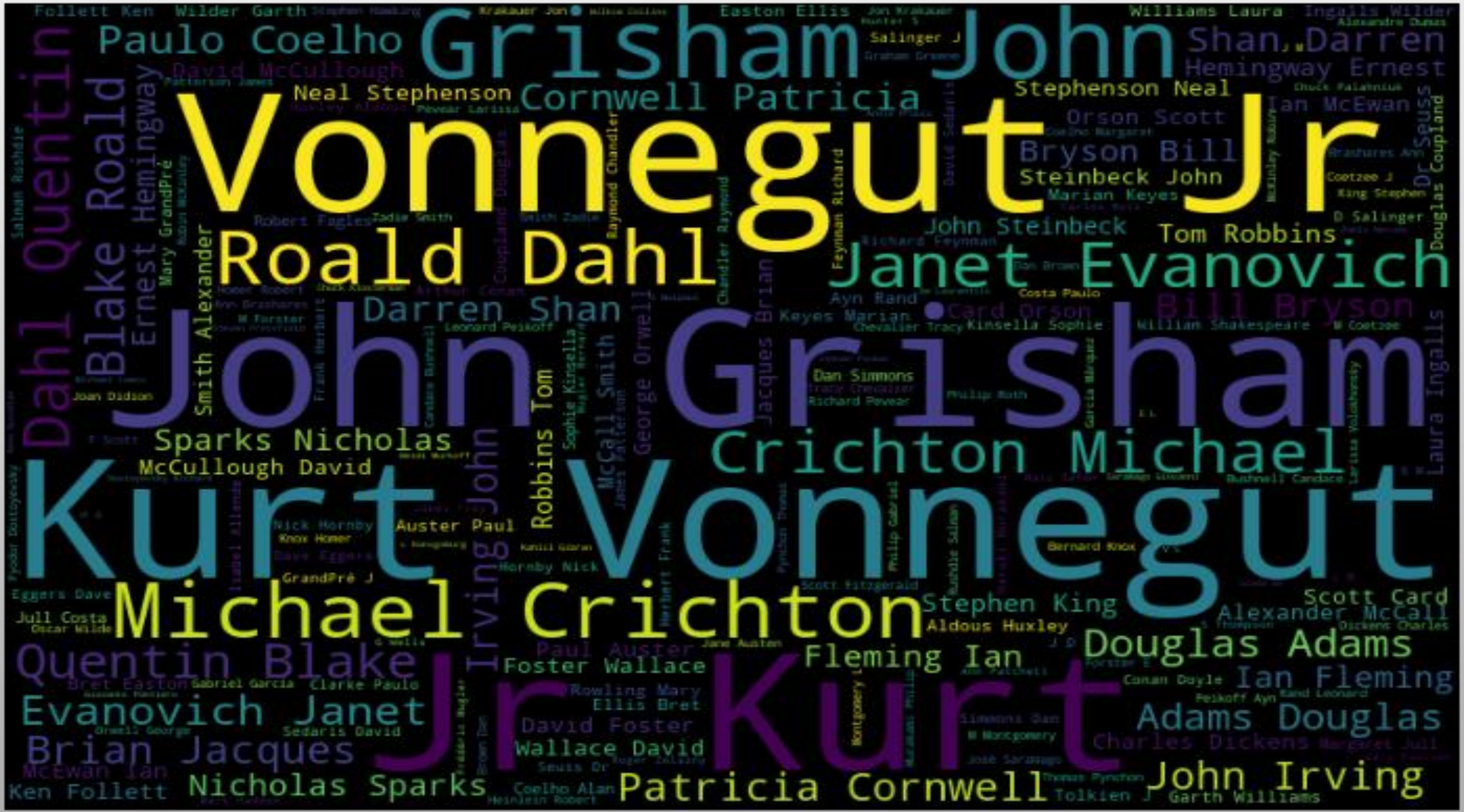
Highly rated authors



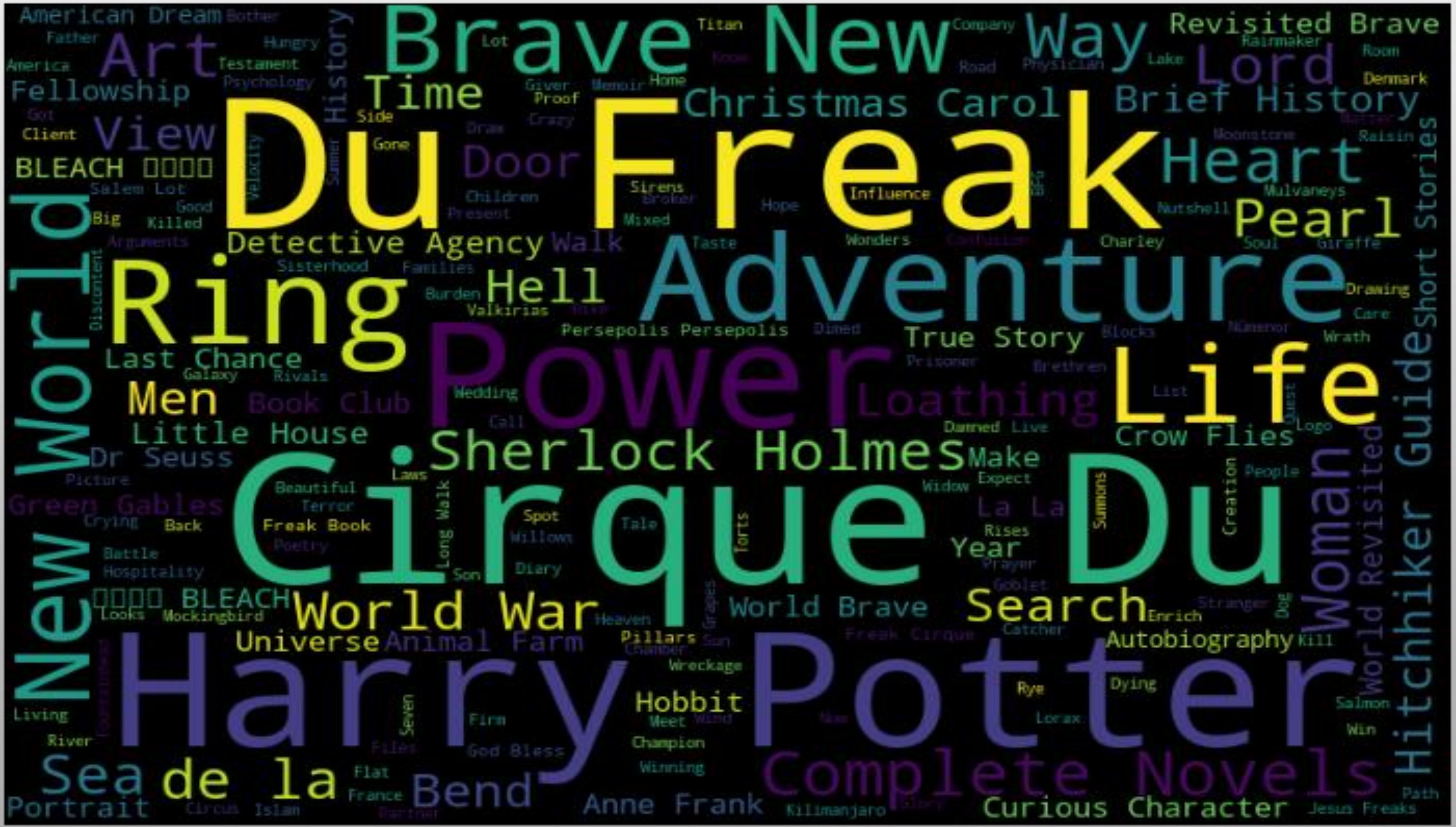
Top author frequency of the book



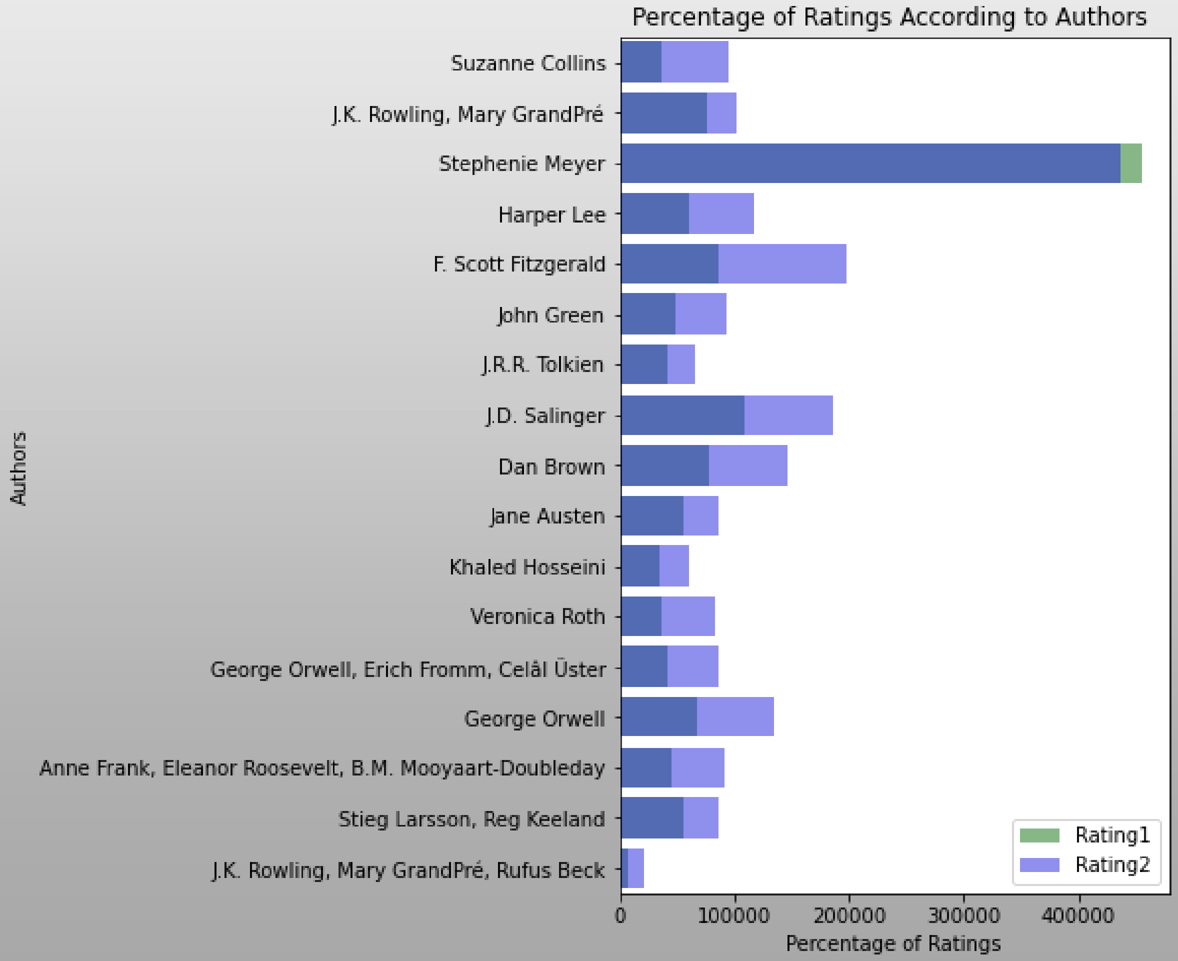
WordCloud of Author Names



WordCloud Of Title of The Books



Percentage of Ratings According to Authors

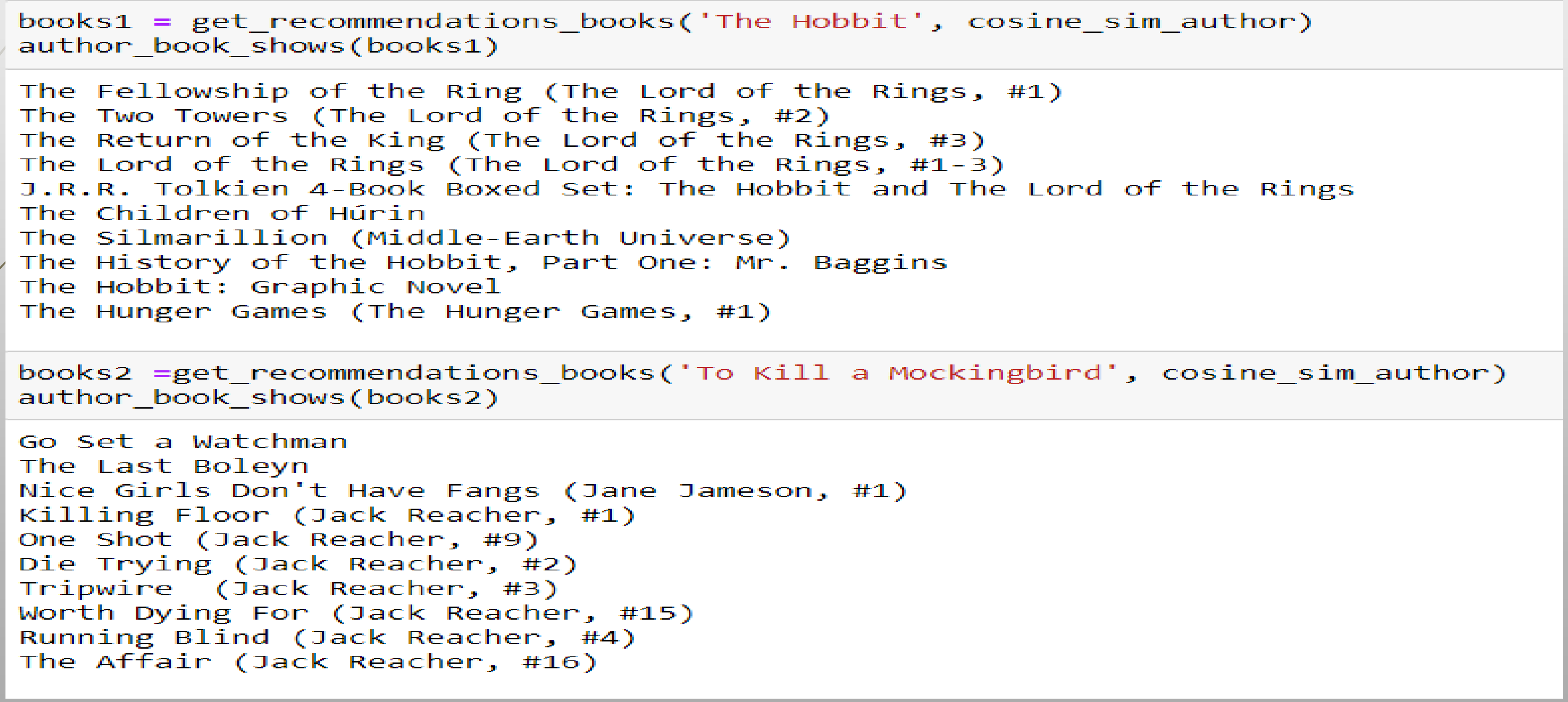


Methods and Algorithms

Methods:

1. Content Based
2. Collaborative Filtering

Content Based



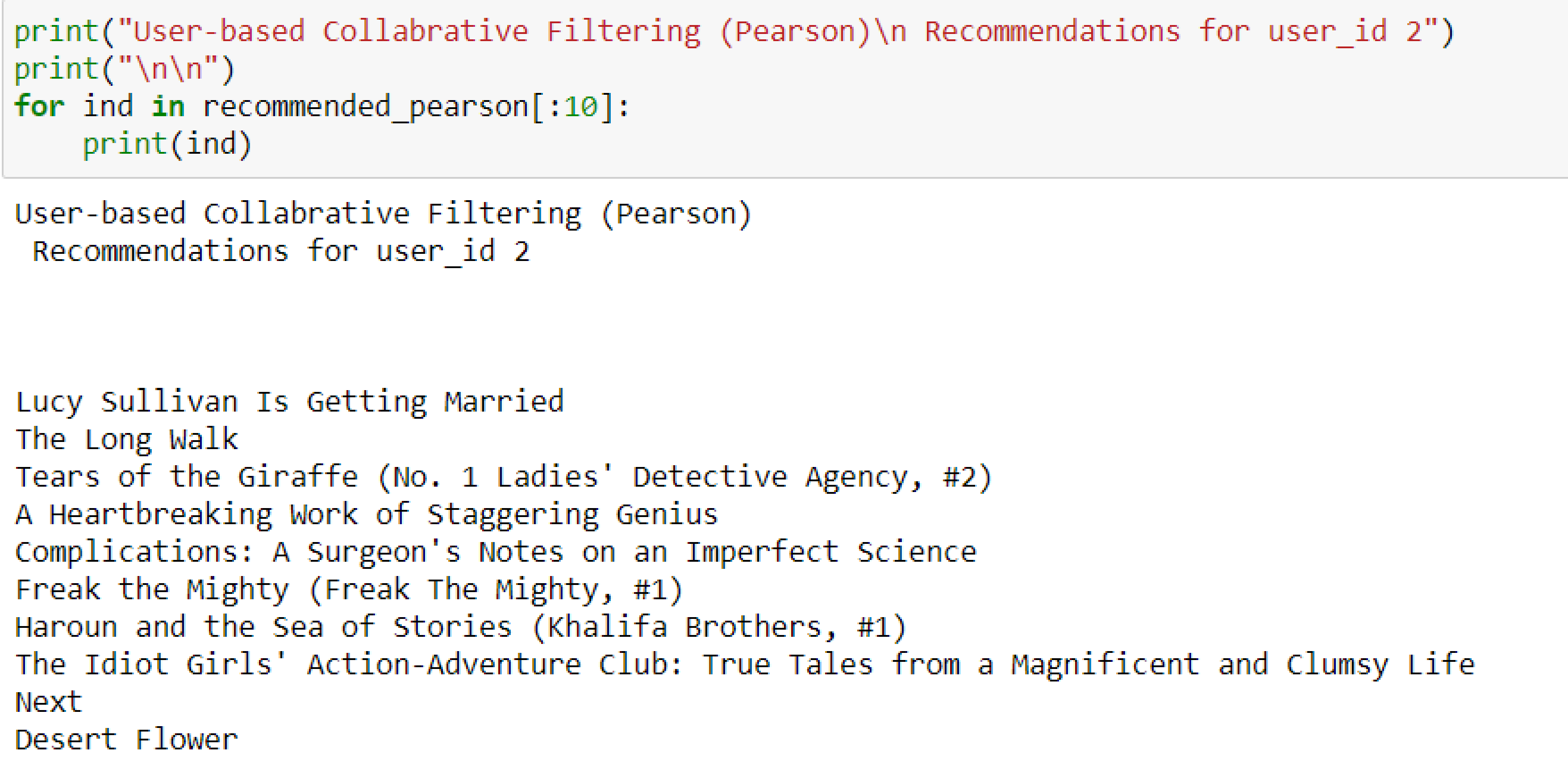
```
books1 = get_recommendations_books('The Hobbit', cosine_sim_author)
author_book_shows(books1)
```

```
The Fellowship of the Ring (The Lord of the Rings, #1)
The Two Towers (The Lord of the Rings, #2)
The Return of the King (The Lord of the Rings, #3)
The Lord of the Rings (The Lord of the Rings, #1-3)
J.R.R. Tolkien 4-Book Boxed Set: The Hobbit and The Lord of the Rings
The Children of Húrin
The Silmarillion (Middle-Earth Universe)
The History of the Hobbit, Part One: Mr. Baggins
The Hobbit: Graphic Novel
The Hunger Games (The Hunger Games, #1)
```

```
books2 =get_recommendations_books('To Kill a Mockingbird', cosine_sim_author)
author_book_shows(books2)
```

```
Go Set a Watchman
The Last Boleyn
Nice Girls Don't Have Fangs (Jane Jameson, #1)
Killing Floor (Jack Reacher, #1)
One Shot (Jack Reacher, #9)
Die Trying (Jack Reacher, #2)
Tripwire  (Jack Reacher, #3)
Worth Dying For (Jack Reacher, #15)
Running Blind (Jack Reacher, #4)
The Affair (Jack Reacher, #16)
```

User Based Collaborative Filtering



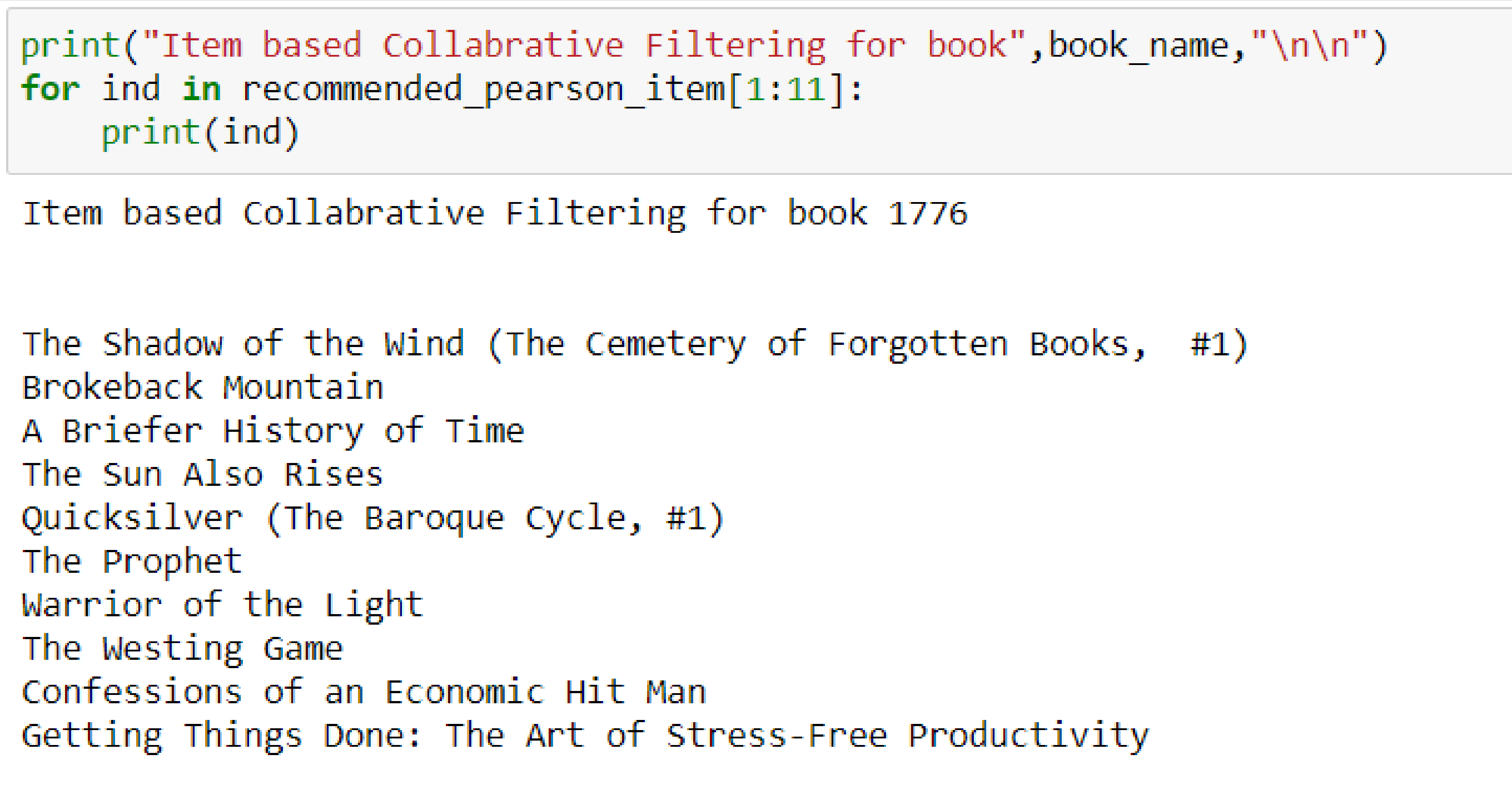
```
print("User-based Collabrative Filtering (Pearson)\n Recommendations for user_id 2")
print("\n\n")
for ind in recommended_pearson[:10]:
    print(ind)
```

```
User-based Collabrative Filtering (Pearson)
 Recommendations for user_id 2



Lucy Sullivan Is Getting Married
The Long Walk
Tears of the Giraffe (No. 1 Ladies' Detective Agency, #2)
A Heartbreaking Work of Staggering Genius
Complications: A Surgeon's Notes on an Imperfect Science
Freak the Mighty (Freak The Mighty, #1)
Haroun and the Sea of Stories (Khalifa Brothers, #1)
The Idiot Girls' Action-Adventure Club: True Tales from a Magnificent and Clumsy Life
Next
Desert Flower
```

Item Based Collaborative Filtering

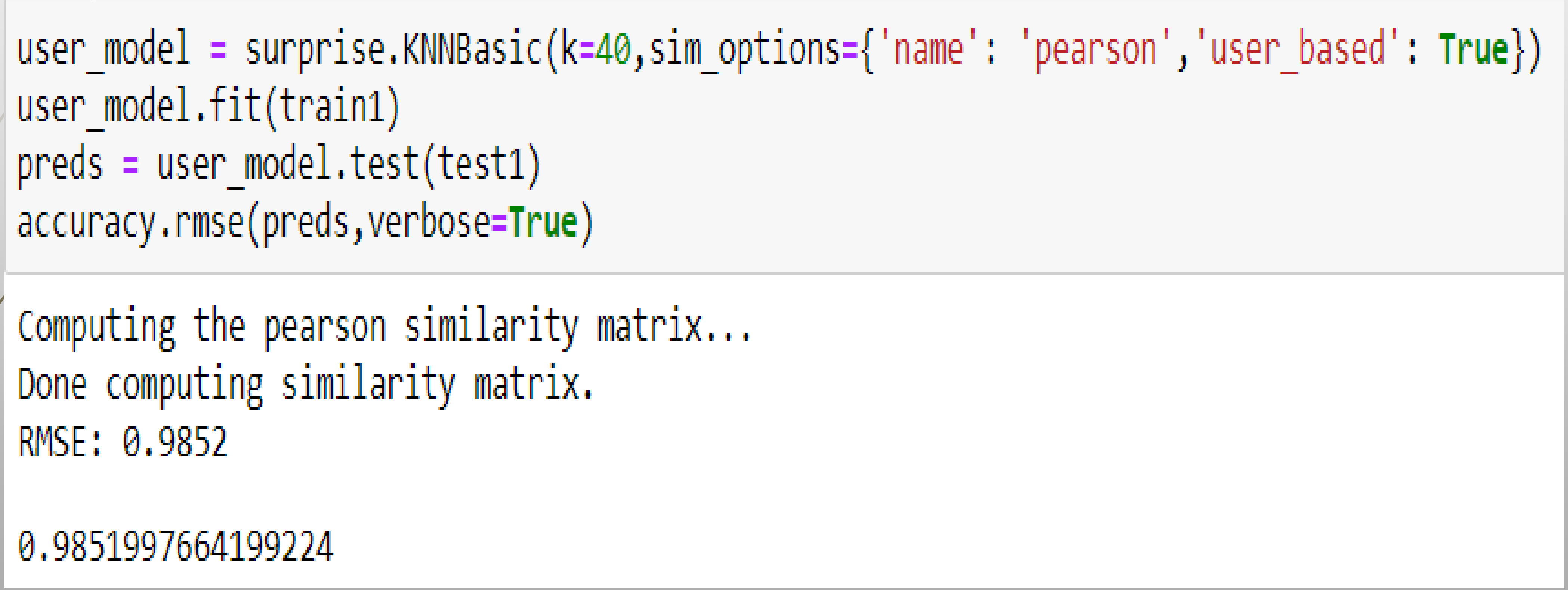


```python
print("Item based Collabrative Filtering for book",book_name,"\n\n")
for ind in recommended_pearson_item[1:11]:
    print(ind)
```

```
Item based Collabrative Filtering for book 1776


The Shadow of the Wind (The Cemetery of Forgotten Books,  #1)
Brokeback Mountain
A Briefer History of Time
The Sun Also Rises
Quicksilver (The Baroque Cycle, #1)
The Prophet
Warrior of the Light
The Westing Game
Confessions of an Economic Hit Man
Getting Things Done: The Art of Stress-Free Productivity
```

Accuracy : User Based

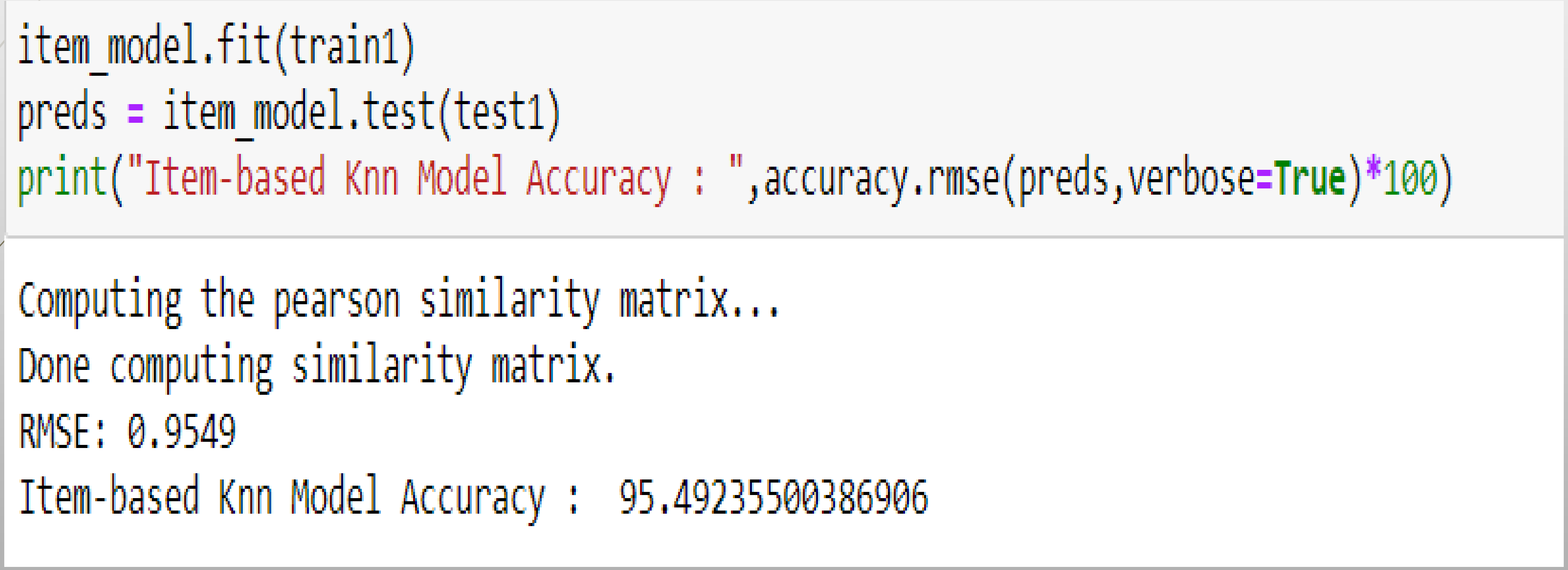


```
user_model = surprise.KNNBasic(k=40,sim_options={'name': 'pearson','user_based': True})
user_model.fit(train1)
preds = user_model.test(test1)
accuracy.rmse(preds,verbose=True)
```

```
Computing the pearson similarity matrix...
Done computing similarity matrix.
RMSE: 0.9852

0.985199764199224
```

Accuracy : Item Based



```
item_model.fit(train1)
preds = item_model.test(test1)
print("Item-based Knn Model Accuracy : ",accuracy.rmse(preds,verbose=True)*100)
```

```
Computing the pearson similarity matrix...
Done computing similarity matrix.
RMSE: 0.9549
Item-based Knn Model Accuracy :  95.49235500386906
```

Conclusion

- In our project we have improvised and modified the recommendation systems. This Book Recommendation System has considered many parameters like ratings, book name, author name, image links etc. We successfully implemented and found the similar books using cosine similarity. Also, the recommendation system was implemented using book dataset and ratings dataset and KNN algorithm to display most similar books based on User based and Item based Collaborative Filtering.

THANK YOU!