LOVELY PROFESSIONAL UNIVERSITY

TERM PAPER

Topic : *A Collaborative Filtering Approach to Kindle Book Recommendations*

NAME : Punam Kumari

REG. NO. : 12323153

ROLL NO. : 30

A Collaborative Filtering Approach to Kindle Book Recommendations

Punam Kumari , Navpreet Kaur
School of computer science
Lovely Professional University, Phagwara, Punjab

Abstract:
It is becoming a very difficult task for the users to select the appropriate books for a specific topic as there were a lot of choices available. There is a need for a system which takes user preferences into consideration while searching and recommending online books to the user. So the objective of this research work is to design an application that recommends books based on users ratings. The system being proposed uses machine learning algorithm like collaborative filtering [CF] that first construct the user-item interaction matrix, then construct vector matrix using cosine similarity measure from user-item interaction matrix and then find the similarity between the books using vector matrix and recommend the top n books similar to the book given by the user as input to the algorithm. The results indicate that recommendation performance is better when both average ratings and cosine vector similarity measure is used as compared to the existing systems.

Introduction

Recommendation system used to suggest items to users based on criteria like past purchases, search history, and other factors. Recommendation system finds items based on user preference and solve the problem of information overloading. It enables the user by providing personalized services and user based content and saves a lot of time and money. In 2009 Netflix launch a competition to improve the accuracy of its movie recommender system by 10%. The recommendation system is very much important in increasing the revenue generation of a company. As per the statistics 35 percent of consumers purchase on Amazon and more than a half of what they watch on Netflix come from recommendations systems. Usually, suggestions from friends and family are always useful to read books or watch anything new but even after looking through all suggestions we may not find something of our preferences, Hence there is a need for a system that takes our preferences into consideration before recommending anything. We do not want to spend time searching for books that we prefer so, we create recommendation tool using collaborative filtering where users can give the name of the book as input and items like the input item are suggested. We implemented two methods of recommendation a popularity-based recommendation using total rating and average ratings of users and a collaborative filtering algorithm by applying cosine vector similarity which measure dot product of two vectors (Book data). Cosine similarity is used to measure similarities in book dataset and find books with high similarities with given book. This recommendation can also be used to recommend movies, articles, music, and news.

2. Related Work

Recommendation systems have become ubiquitous across various domains, including movies, books, research articles, and social tags. They typically fall into three main categories: collaborative filtering, content-based filtering, and hybrid recommendation.

Collaborative filtering relies on analyzing user behavior, activities, or preferences to predict what users might like based on similarities with other users. This approach doesn't require specialized knowledge and improves as users engage more, but it can face challenges like data sparsity. Collaborative filtering can be divided into two types: item-based and user-based, where similarities are calculated using metrics such as cosine, Euclidean, or Jaccard.

Content-based filtering, on the other hand, focuses on item descriptions and user preferences profiles. By recommending items similar to those a user has liked before, it bypasses some of the challenges of collaborative filtering.

Hybrid recommendation systems combine elements of both collaborative and content-based filtering. By leveraging the strengths of each approach, they can mitigate issues like cold start and data sparsity, offering more robust recommendations.

3. Existing System

1. Okon et.al. (2018) proposed a model that generates recommendations to buyers, through an enhanced CF algorithm, a quick sort algorithm and Object-Oriented Analysis and Design Methodology (OOADM). Scalability was ensured through the implementation of Firebase SQL. This system performed well on the evaluation metrics.
2. Kurmashov et.al. (2015) used Pearson correlation coefficient-based CF to provide internet based recommendations to book readers and evaluated the system through an online survey.
3. Mathew et.al. (2016) proposed a system that saves details of books purchased by the user. From these Book contents and ratings, a hybrid algorithm using collaborative filtering, content-based filtering and association rule generates book recommendations. Rather than Apriori, they recommended the use of Equivalence class Clustering and bottom-up Lattice Transversal (ECLAT) as this algorithm International Journal of Engineering Research & Technology (IJERT) http://www.ijert.org ISSN: 2278-0181 IJERTV12IS040195 (This work is licensed under a Creative Commons Attribution 4.0 International License.) Published by : www.ijert.org Vol. 12 Issue 04, April-2023 365 is faster due to the fact that it examines the entire dataset only once.
4. Parvatikar et.al. (2015) proposed item-based collaborative filtering and association rule mining to give recommendations. Similarity between different users was computed through Adjusted Cosine Vector Similarity function. Better recommendations were obtained as through this method data sparsity problem was removed.
5. Ayub et.al. (2018) proposed a similarity function like Jaccard Similarity to locate alike items and users for the enquiring item and user in nearest neighbor based collaborative filtering. They proposed that absolute value of ratings should be taken as against the ratio of co-rated items taken in Jaccard Similarity. They also compared performance of their method with other similarity measures.
6. Gogna and Majumdar suggested the use of buyer’s demographic and item category to overcome data sparsity and cold start problems in their movie recommendation system. Latent Factor Model (LFM) was used. They developed a matrix to match the

buyer and user information to get a dense user and dense item matrix. Label Consistency map, the outcome of this system was used to suggest unrated and other items to new buyers.

7. Chatti et.al. (2013) suggested tag-based and rating based CF recommendation in technology enhanced learning (TEL) to resolve the data sparsity problem and extract relevant information from the rating database. Memory and model oriented 16 varied tag-based Collaborative filtering algorithms were evaluated for buyer satisfaction and accuracy of recommendations in Personal Learning Environments
8. Choi et.al. (2010) proposed RS based on HYRED, a hybrid algorithm using both content and collaborative filtering on a compact dataset (by reducing user interest items) and neighbor data. HYRED used altered Pearson Coefficient based Collaborative filtering and distance-toboundary (DTB) Content filtering. This would result in better and faster recommendation for large amount of data.
9. Liu et.al. (2012) added the dimension of user-interest. They proposed iExpand, a 3-tier model i.e. user, userinterest and item. This helped in overcoming the issues of overspecialization and cold start as well as reducing computation costs.
10. Feng et.al. (2018) proposed a RS for movies based on a similarity model constituted of factors S1 (similarity between users), S2 (ratio of co-rated items) and S3 (user's rating choice weight). This RS was particularly useful for sparse datasets.
11. According to Aggarwal , C.C., Collaborative Filtering method is used in recommendation systems to develop recommendations based on ratings provided by the other users of the system. If buyer's ratings of items match, it is likely that their ratings of other items will also match, this is the basic assumption of CF. Computers cannot gauge qualitative factors such as taste or quality, therefore recommendations based on the ratings of humans who can rate on the basis of qualitative factors, i.e., collaboration, will give a better outcome.
12. Gattu Vijaya Kumar, Prasanta Kumar Sahoo, K.Eswaran explain the main reason we need a recommendation system in the current generation is because humans have extremely many alternatives to utilize required information which is popular from the Internet. It implements five classification algorithms such as Support-Vector Machines, Logistic Regression, Multinomial Naive Bayes, Multilayer Perceptron and K-Nearest Neighbors .It was observed that from the comparison of all the algorithms Support-Vector Machine gives 75.13% accuracy.
13. Prasanta kumar sahoo, Kodaty Sri Sai Chaitany and N. Archana suggested most of the e-commerce retailers are suffering, in displaying the targeted and relevant results

for a search keyword given by customers. In this scenario analyzing their past interests and recent behavior of the customers is one of the important aspects to confirm the user relevant search results. The online customer behavior has been analyzed by using Homophily Detection Algorithm in this research work. They implements Behavior analysis, Trend analysis and Personalization on customer data and displaying relevant search results when a customer search for a particular product which leads to greater customer satisfaction. Literature survey suggested that recommendation systems are being used by large number of online marketers to increase their sales by offering products to customers which match their tastes

4. Proposed System

This research work mainly focuses on popularity based method and Collaborative Filtering Technique for the recommendation system. Collaborative filtering [CF] technique primarily focuses on the relationship between users and items. It is a technique that can filter out items that a user might like on the basis of ratings given by the other users and recommends the top-n similar items to the user. The similarity of items is determined by the similarity of the ratings of those items by the users who have rated both items. Item-based collaborative filtering is being used by Amazon for customers. In an existing system where there are more users than items, item-based filtering is faster and more stable than user-based. This algorithm uses a similarity measure to find similarity between items.

5. Methodology

1. Import all necessary libraries like pandas, numpy, seaborn etc

2. Data Preprocessing, in this step the dataset is checked for missing values and null values in the data and remove or fill them.

3. The design of the recommendation system based on two methods as given below:

a. Popularity based: To implement popularity based recommendation, the data set is selected by merging the two data frames. One is data frames for no of ratings and the other one is average ratings and merge them. Popular books are those whose average rating is more than 15 ratings are selected as per the criteria and top 5000 results are printed.

b. collaborative filtering: To implement recommendation based on collaborative filtering, cosine similarity is being used to find the similarity of user preferences and recommend books based on cosine similarity of book.

6. Overview of Dataset

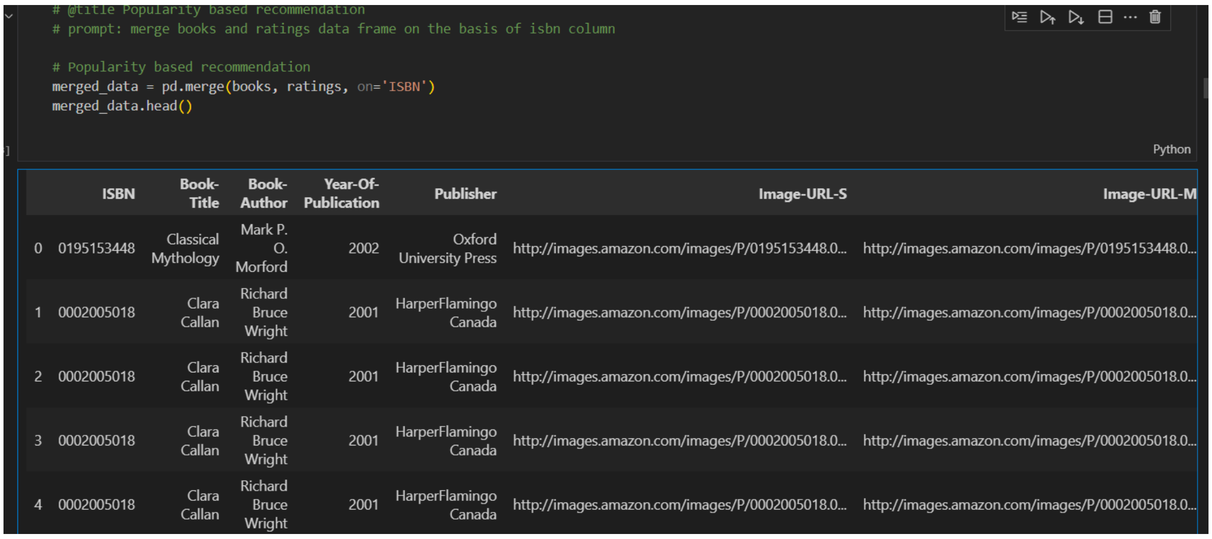
6.1 Importing Dataset

Datasets are critical component of AI development because they provide the training data that is used to train and test machine learning models. In this project Datasets such as Books.csv, Ratings.csv and Users.csv was collected from Kaggle for the implementation purpose.

6.2 Merging Dataset

The data set is taken by merging Books.csv and Rating.csv to form merged_data. The data contains seven attributes and in the process unnecessary attributes are removed. The attributes of Bookrec.csv are: User-ID, ISBN, Book-Rating, Book-Title, Book-Author, Year-OfPublication.

```
merged_data = pd.merge(books, ratings, on='ISBN')

merged_data.head()
```



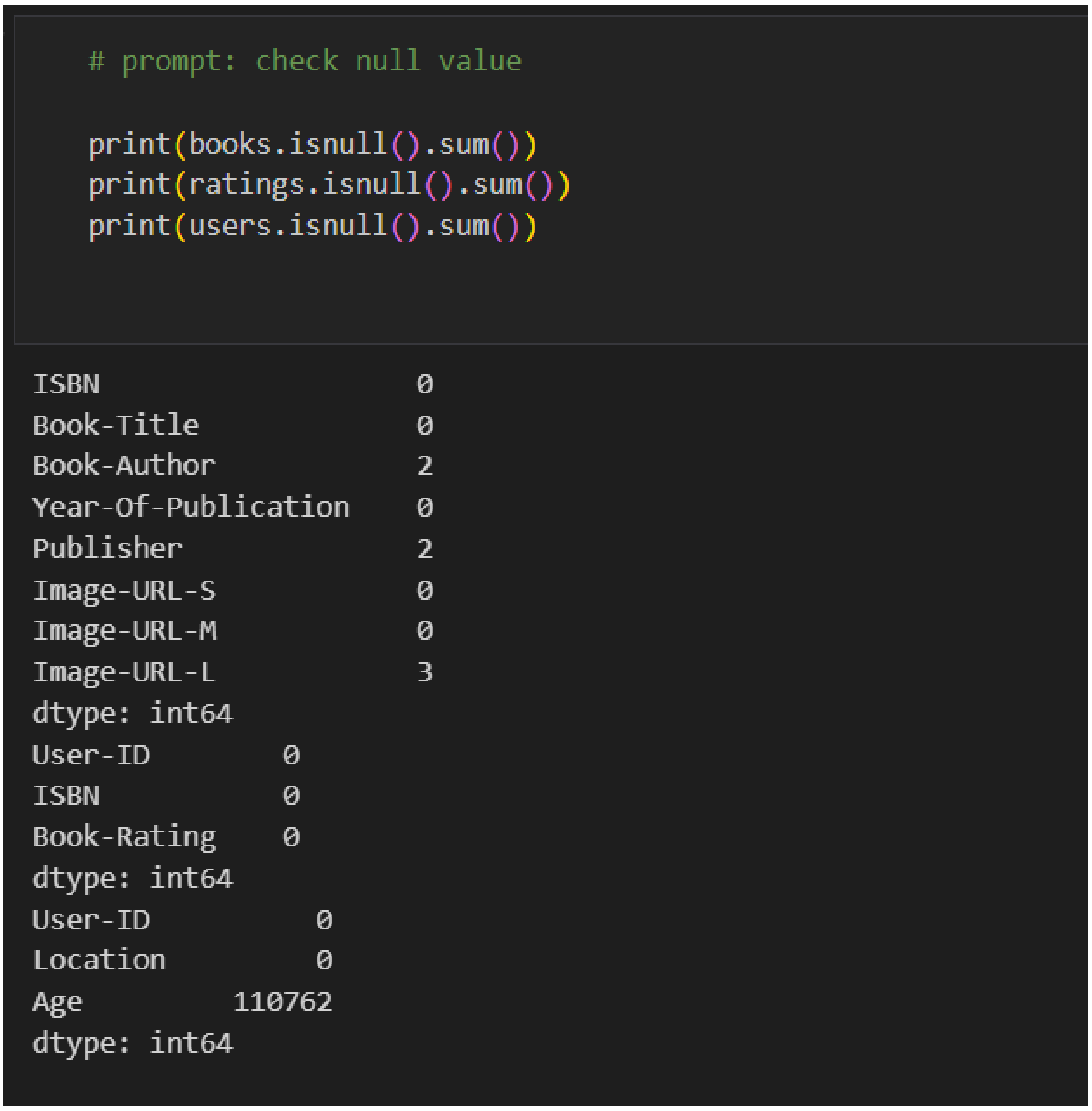
```
# @title Popularity based recommendation
# prompt: merge books and ratings data frame on the basis of isbn column

# Popularity based recommendation
merged_data = pd.merge(books, ratings, on='ISBN')
merged_data.head()
```

Python

	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	Image-URL-M
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com/images/P/0195153448.0...
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...
2	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...
3	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...
4	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com/images/P/0002005018.0...

6.3 Data Cleaning

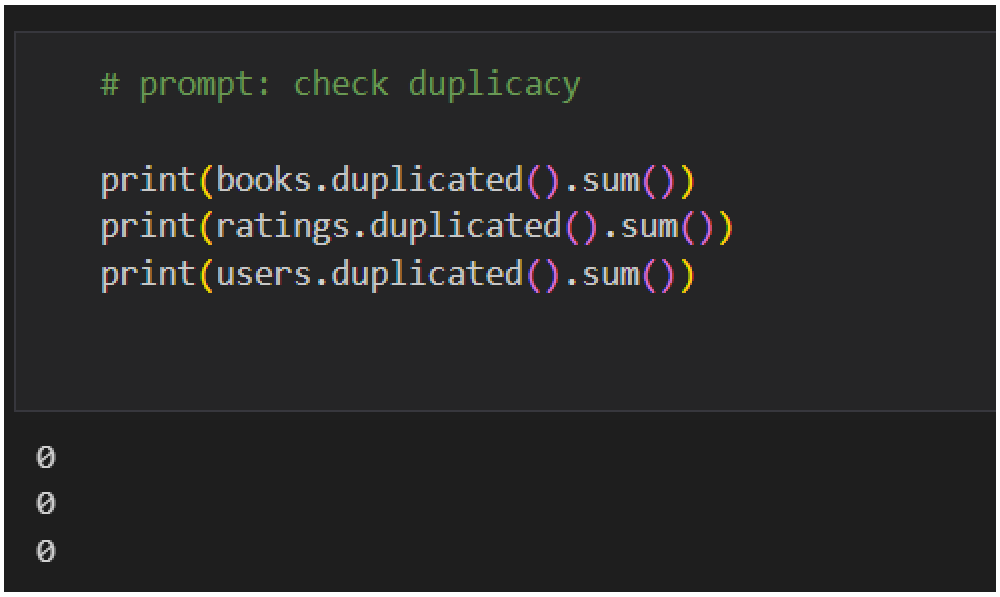
In data cleaning step the dataset is analyzed and checked for null values, missing values, and duplicate values The goal of data cleaning is to ensure that the data is accurate, consistent, and free of errors. It is important to clean data as leaving them can negatively impact ML models. This can be done using the pandas.



```
# prompt: check null value

print(books.isnull().sum())
print(ratings.isnull().sum())
print(users.isnull().sum())
```

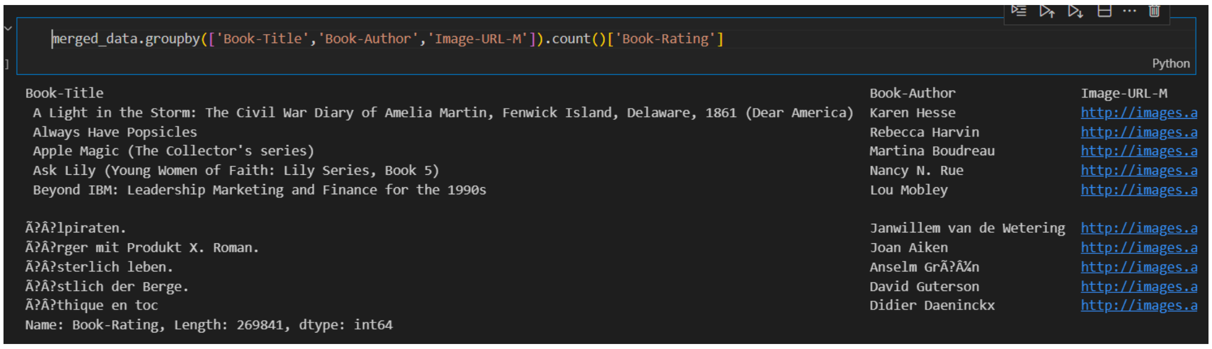
```
ISBN                   0
Book-Title             0
Book-Author            2
Year-Of-Publication    0
Publisher              2
Image-URL-S            0
Image-URL-M            0
Image-URL-L            3
dtype: int64
User-ID        0
ISBN           0
Book-Rating    0
dtype: int64
User-ID          0
Location         0
Age         110762
dtype: int64
```

checking for duplicacy



```python
# prompt: check duplicacy

print(books.duplicated().sum())
print(ratings.duplicated().sum())
print(users.duplicated().sum())
```

```
0
0
0
```



```python
merged_data.groupby(['Book-Title','Book-Author','Image-URL-M']).count()['Book-Rating']
```

```
Book-Title                                                                                    Book-Author               Image-URL-M
 A Light in the Storm: The Civil War Diary of Amelia Martin, Fenwick Island, Delaware, 1861 (Dear America)  Karen Hesse               http://images.a
 Always Have Popsicles                                                                        Rebecca Harvin            http://images.a
 Apple Magic (The Collector's series)                                                         Martina Boudreau          http://images.a
 Ask Lily (Young Women of Faith: Lily Series, Book 5)                                         Nancy N. Rue              http://images.a
 Beyond IBM: Leadership Marketing and Finance for the 1990s                                   Lou Mobley                http://images.a

Ã?Â?lpiraten.                                                                                Janwillem van de Wetering http://images.a
Ã?Â?rger mit Produkt X. Roman.                                                               Joan Aiken                http://images.a
Ã?Â?sterlich leben.                                                                          Anselm GrÃ?Â¼n            http://images.a
Ã?Â?stlich der Berge.                                                                        David Guterson            http://images.a
Ã?Â?thique en toc                                                                            Didier Daeninckx          http://images.a
Name: Book-Rating, Length: 269841, dtype: int64
```

6.4 Cosine vector similarity (CVS):

Cosine similarity is measured by the cosine of angle between vectors and finds, if they point same direction. It measures cosine angle by using dot product of vectors. It measures similarity between two vectors. It is represented by:

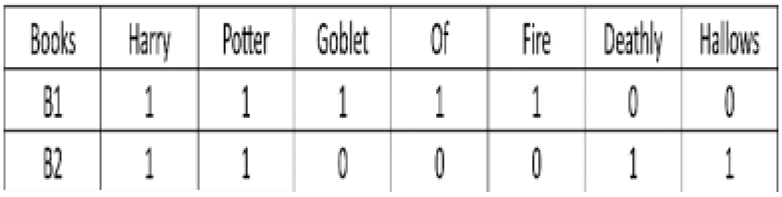
$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Case study 1

Here,

A=First vector B=Second vector

Example: B1= Harry potter: Goblet of fire B2=Harry potter: Deathly hallows After creating a word table from the Books, Books can be represented by the following vectors:



Books	Harry	Potter	Goblet	Of	Fire	Deathly	Hallows
B1	1	1	1	1	1	0	0
B2	1	1	0	0	0	1	1

B1= {1,1,1,1,1,0,0} B2= {1,1,0,0,0,1,1} Using these two vectors we can calculate cosine similarity. First, we calculate the dot product of the vectors:

B1. B2= 1.1+1.1+1.0+1.0+1.0+0.1+0.1 =2

Magnitude of vectors: ||B1||= √5 ||B2||= √4

Cosine similarity (B1, B2) =2/ √5. √4 =0.44721

Cosine similarity implementation

1. Create a matrix showing similarities between books in dataset.

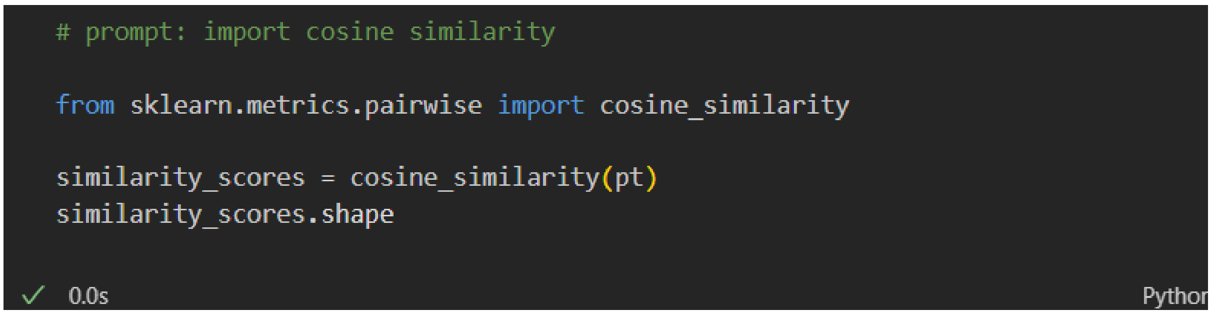
2. Import cosine similarity from sklearn library and create cosine matrix using similarities in dataset.

3. Find books with high similarity score with the book given by user.

4. Recommend books with high similarities with the book given by user.

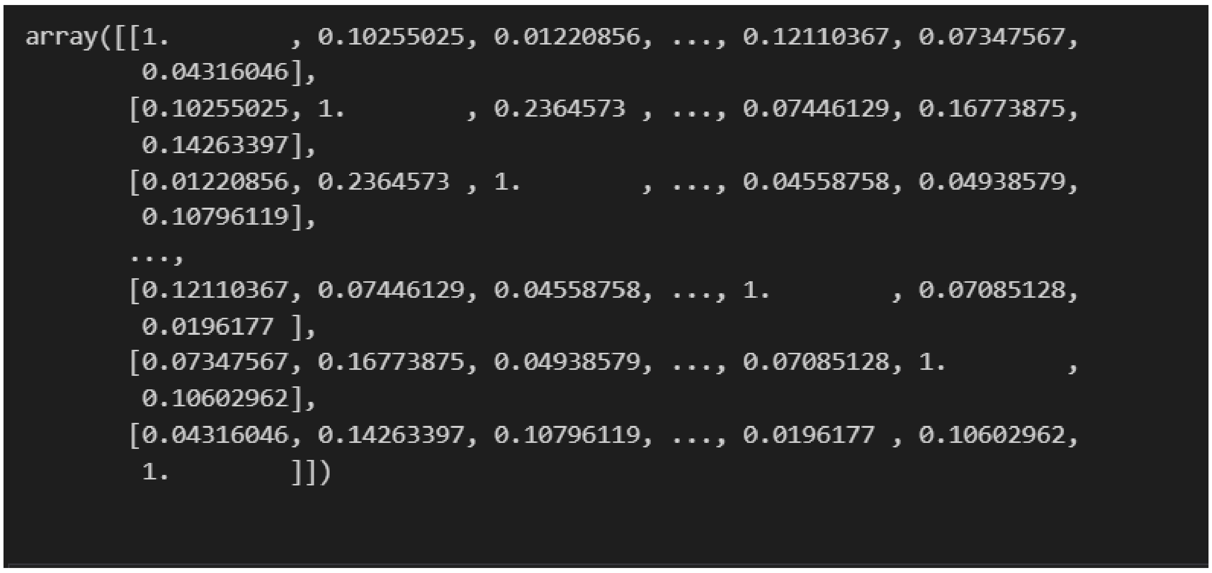
Data similarity matrix

6.4 Cosine similarity importing and implementation:

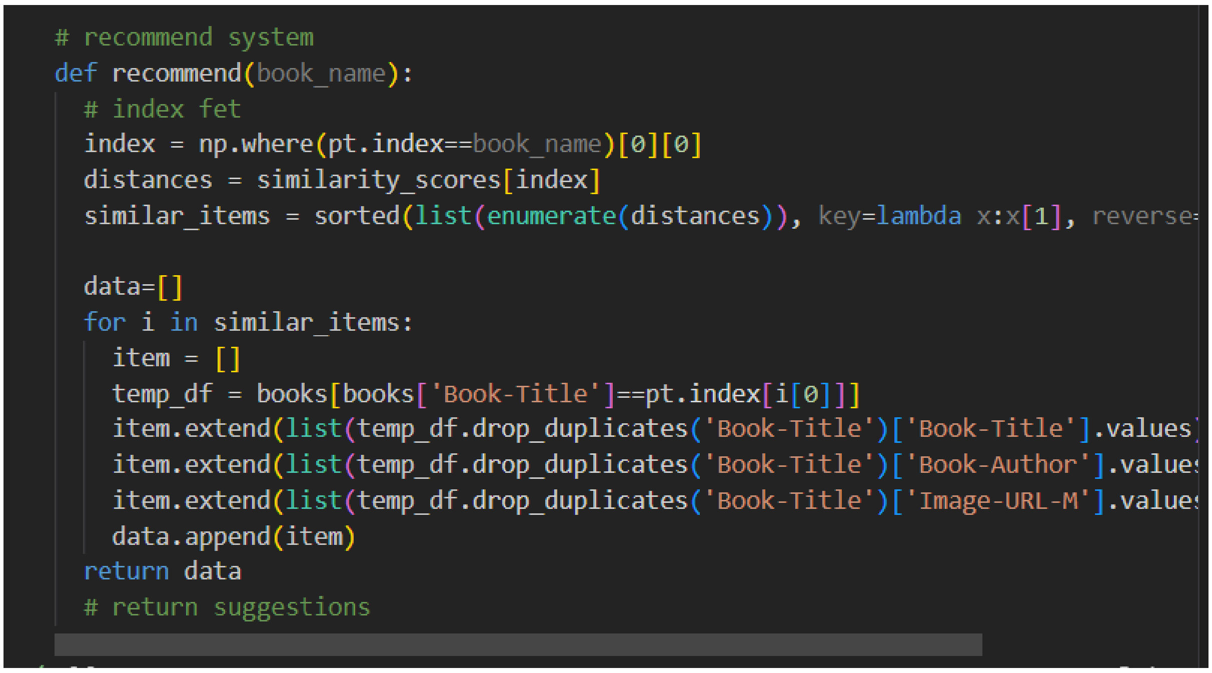


```python
# prompt: import cosine similarity

from sklearn.metrics.pairwise import cosine_similarity

similarity_scores = cosine_similarity(pt)
similarity_scores.shape
```

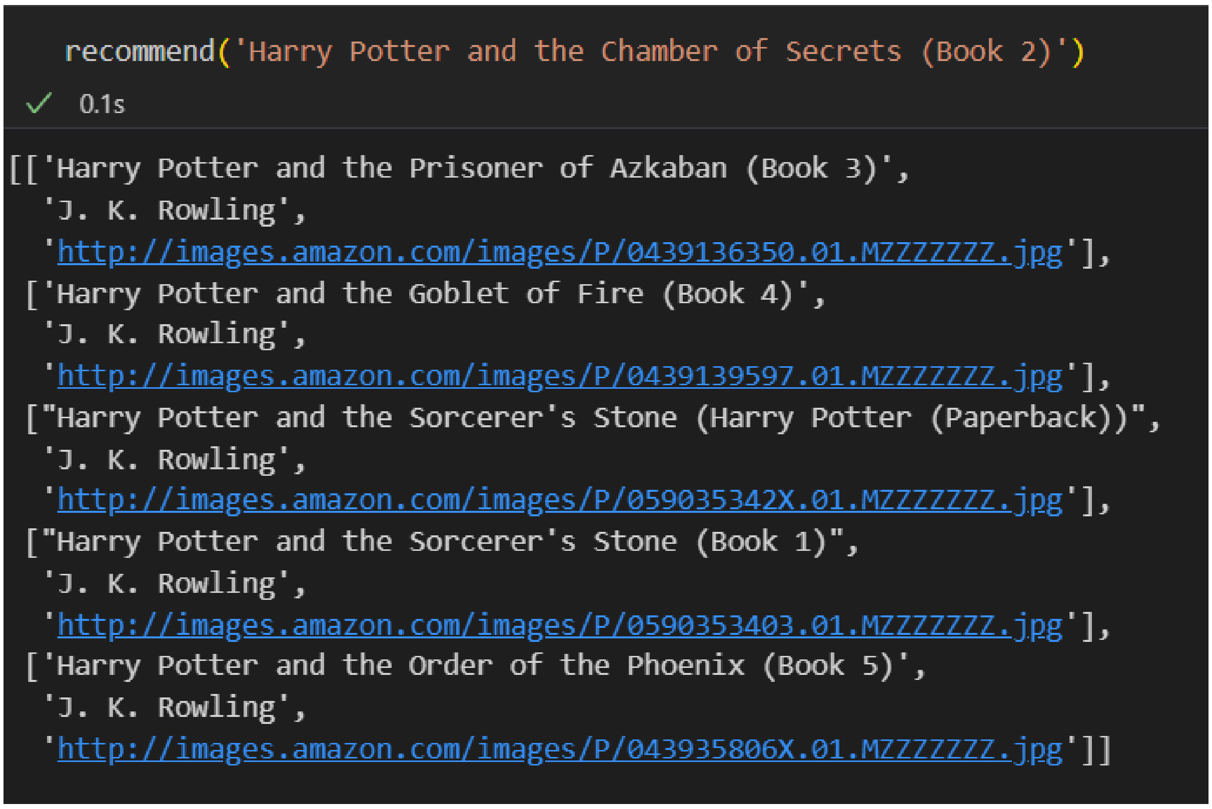
✓ 0.0s Pythor



```
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],
       ...,
       [0.12110367, 0.07446129, 0.04558758, ..., 1.        , 0.07085128,
        0.0196177 ],
       [0.07347567, 0.16773875, 0.04938579, ..., 0.07085128, 1.        ,
        0.10602962],
       [0.04316046, 0.14263397, 0.10796119, ..., 0.0196177 , 0.10602962,
        1.        ]])
```



```
# recommend system
def recommend(book_name):
    # index fet
    index = np.where(pt.index==book_name)[0][0]
    distances = similarity_scores[index]
    similar_items = sorted(list(enumerate(distances)), key=lambda x:x[1], reverse=

    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'].value
        item.extend(list(temp_df.drop_duplicates('Book-Title')['Image-URL-M'].value
        data.append(item)
    return data
    # return suggestions
```



```
recommend('Harry Potter and the Chamber of Secrets (Book 2)')
✓ 0.1s
[['Harry Potter and the Prisoner of Azkaban (Book 3)',
  'J. K. Rowling',
  'http://images.amazon.com/images/P/0439136350.01.MZZZZZZZ.jpg'],
 ['Harry Potter and the Goblet of Fire (Book 4)',
  'J. K. Rowling',
  'http://images.amazon.com/images/P/0439139597.01.MZZZZZZZ.jpg'],
 ["Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))",
  'J. K. Rowling',
  'http://images.amazon.com/images/P/059035342X.01.MZZZZZZZ.jpg'],
 ["Harry Potter and the Sorcerer's Stone (Book 1)",
  'J. K. Rowling',
  'http://images.amazon.com/images/P/0590353403.01.MZZZZZZZ.jpg'],
 ['Harry Potter and the Order of the Phoenix (Book 5)',
  'J. K. Rowling',
  'http://images.amazon.com/images/P/043935806X.01.MZZZZZZZ.jpg']]
```

5. Conclusion

Recommendation systems are very popular in e-commerce applications such as online book store and can significantly rise the company's revenue generation. Recommendation system helps us to lighten the information overloading problem where the user will have a lot of choices and not able to understand perfectly what to buy or what to see. Recommendation system solves this problem and provides users with personalized content after searching a large volume of information. The proposed work to design a recommendation system based on popularity method and collaborative filtering is much more accurate than the existing

systems.

6. References

1. ResearchGate Prasanta Kumar Sahoo Anurag University · Department of Computer Science & Engineering Doctor of Engineering
2. ChatGPT
3. Google Scholar