Institute of Technology of Cambodia
Department of Applied Mathematics and Statistics

Book Recommendation System

Name	ID
PHETH Soriyuon	e20210674
PHO Rotha	e20211543
PHOEUN Rajame	e20211748
PHOEURN Kimhor	e20210823
SIV Lyheng	e20211734

Lecturer : Dr. PHAUK Sokkhey

Academic Year 2023 - 2024

Contents

I Introduction to Recommendation System

Recommendation system is a type of algorithm that provides personalized suggestions or recommendations to users.
These suggestions are typically related to items or content that the user might find interesting or relevant, such as products, movies, music, and books.

II Objective of the Project

- User Satisfaction : providing accurate and relevant book recommendations that align with their reading preferences.
- Sales and Revenue Growth : recommending books that users are likely to purchase which will increase the sales.
- Aiding New Readers : help new readers choose what to read.

III About the Datasets

The datasets are from Kaggle.com, but originally they are collected from Amazon.
1. Books (271360 rows $\times$ 8 columns)

- Book Title
- Book-Author
- ISBN : is a national and international standard identification number for uniquely identifying books.
- Year-Of-Publication : Year of the books when published.
- Publisher : a person or corporation whose business is publishing.
- Image-URL-S : link to book cover of size S
- Image-URL-M : link to book cover of size M
- Image-URL-L : link to book cover of size L

2. Ratings (1048575 rows $\times$ 3 columns)

- User-ID : Unique ID of each user.
- Book-rating : Rating from readers from 0 to 10
- ISBN

3. Users (278858 rows × 3 columns)

- Location : Location of Users.
- User-ID
- Age

IV Data Cleaning

In this process, We want to repair, remove, and replace the incorrect, corrupted, incorrectly format, duplicate, or incomplete data within the datasets. Moreover, we want to ensure the accuracy of the model that is produced from the data and prevent blindly error in machine learning process.

4.1 Dataset Books

Here are the first 5 rows of Books

```
book.head()
```

	ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher	Image-URL-S	
0	0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press	http://images.amazon.com/images/P/0195153448.0...	http://images.amazon.com
1	0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	http://images.amazon.com/images/P/0002005018.0...	http://images.amazon.com
2	0060973129	Decision in Normandy	Carlo D'Este	1991	HarperPerennial	http://images.amazon.com/images/P/0060973129.0...	http://images.amazon.com
3	0374157065	Flu: The Story of the Great Influenza Pandemic...	Gina Bari Kolata	1999	Farrar Straus Giroux	http://images.amazon.com/images/P/0374157065.0...	http://images.amazon.com
4	0393045218	The Mummies of Urumchi	E. J. W. Barber	1999	W. W. Norton & Company	http://images.amazon.com/images/P/0393045218.0...	http://images.amazon.com

4.1.1 Error Detection In Book-Data

1. Unnecessary columns

```
    Columns = ['Image-URL-S', 'Image-URL-M', 'Image-URL-L']
```

2. Null Value

```
    Column['Book-Author'] = 2; # row=118033 and row=187689
    Column['Publisher'] = 2; # row=128890 and row=129037
```

3. Unknown Author

```
    #Row(118033) (Base on ISBN: Source: Author_detailed = ''Unknown'')
```

4. Wrong datatype and information

```
    Column['Year-Of-Publication'].unique()
    Output: Array = {['DK Publishing Inc', 'Gallimard']}
    # DK Publishing Inc: row=209538 & row=221678
    # Gallimard: row=220731

```

5. Outlier(Year-Of-Publication)

```
    #We set limit range (1800 < 'Year-Of-Publication' < 2023)
    Column ['Year-Of-Publication'] < 1800 #detect 4620 rows
    Column ['Year-Of-Publication'] > 2023 #detect 13 rows
    #To sum up, there are 4633 outliers
```

6. Duplication

```
    #Detect 29225 rows (Duplicate on Title)
```

4.1.2 Handling Error Of Book-Data

1. Unnecessary columns

```
    #Drop the entire column
    Dataframe.drop(columnName, inplace = True)
```

2. Null Value

```
    #Replace with detailed based on ISBN
    Dataframe.loc[NumRow, 'columnName'] = 'NewVariable'
```

3. Unknown Author

```
#Replace it with the 'Unknown' as the ISBN provided
```

4. Wrong datatype and information

```
#Change the datatype to (int)
#Surfing and replace with correct information (Method: Browsing with ISBN)
```

5. Outlier(Year-Of-Publication)

```
1. Try solving it
a. #Find another dataset (BookFromAnotherBrench.csv)
b. #Freeing ISBN and publication-year from new Dataset
c. #Merging it with outlier on ISBN
(Result: 3 rows Matched)
#We try this with a few different datasets and the result is either none or a few matches. So, we are currently unable to solve it with this solution.
2. Final decision
a. #Replace outlier year with: NAN
b. #And override it with a string: 'Not in our system'
```

6. Duplication

```
#Drop all duplication
```

4.2 Dataset Ratings

+ First 5 rows of Ratings

	User-ID	ISBN	Book-Rating
0	276725	034545104X	0
1	276726	0155061224	5
2	276727	0446520802	0
3	276729	052165615X	3
4	276729	0521795028	6

In ratings dataset, it is already cleaned. Therefore, there will be no need for cleaning for this dataset. This datasets contains explicit ratings (1-10) and implicit rating (0). So we take only the explicit ratings as the implicit rating has no use.

4.3 Dataset Users

1. First 5 rows of Users

	User-ID	Location	Age
0	1	nyc, new york, usa	NaN
1	2	stockton, california, usa	18.000000
2	3	moscow, yukon territory, russia	NaN
3	4	porto, v.n.gaia, portugal	17.000000
4	5	farnborough, hants, united kingdom	NaN

2. Checking for NULL Values

```
user.isna().sum()
```

User-ID	0
Location	0
Age	110762

3. Descriptive Statistics

```
user['Age'].describe()
```

	Age
count	168096.000000
mean	34.751434
std	14.428097
min	0.000000
25%	24.000000
50%	32.000000
75%	44.000000
max	244.000000

4. Create a new dataframe that consists of outliers and get only the outliers that have rated books. If not, drop them.

```
filtered_df = user[pd.isnull(user['Age']) | (user['Age'] < 5) | (user['Age'] > 95)]
filterd_age_rate = filtered_df[filtered_df['User-ID'].isin(explicit_rating['User-ID'])]
```

+ Then drop all outliers from the original dataframe.

```
user = user.drop(filtered_df.index)
```

+ Fill the outliers with average age of users

```
filterd_age_rate['Age'] = np.nan
filterd_age_rate.fillna(round(34.739380), inplace=True)
```

+ Connect it back to the original Dataframe.

```
user = pd.concat([user, filterd_age_rate])
```

+ Create a new column that represents that age stage of users

```
def age_group(age):
if age < 13:
x = 'Children'
elif 13 <= age < 18:
x = 'Teenager'
elif 18 <= age <36:
x = 'Adult'
elif 36<= age < 56:
x = 'Middle-Aged Adult'
else:
x= 'Elderly'
return x

user['Age-Group'] = user['Age'].apply(lambda x: age_group(x))
```

+ Extract random symbols out of location column and get only the country

```
for i in user:
user['Country']=user.Location.str.extract(r'\,+\s?(\w*\s?\w*)\"*$')
```

+ Drop Location out of the dataframe

```
user.drop('Location', inplace= True, axis=1)
```

5. Final Dataset

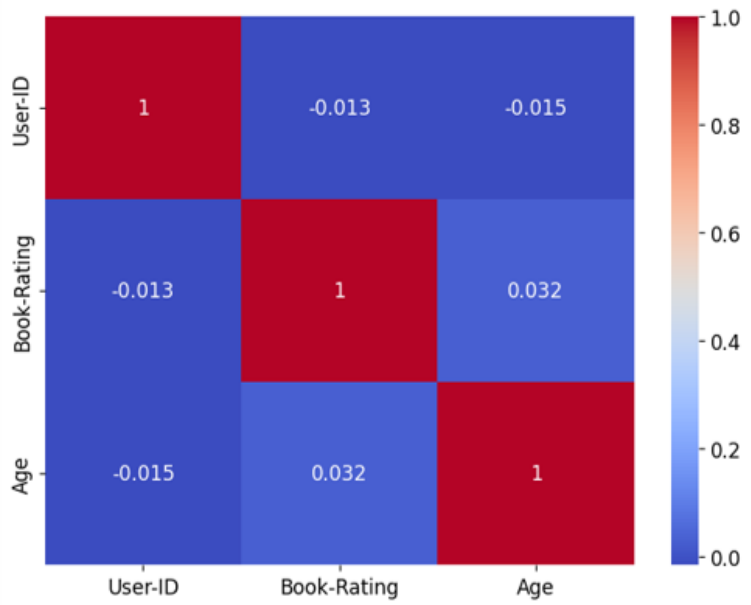
	User-ID	Age	Age-Group	Country
1	2	18.000000	Adult	USA
3	4	17.000000	Teenager	PORTUGAL
5	6	61.000000	Elderly	USA
9	10	26.000000	Adult	SPAIN
10	11	14.000000	Teenager	AUSTRALIA

+ Merging the Dataset

```
df = pd.merge(book, explicit_rating, on='ISBN', how='inner')
df = pd.merge(df, user, on='User-ID', how='inner')
df.isna().sum()
# Since there are NaN values in Country, so fill with "UNKNOWN"
df.fillna("UNKNOWN", inplace=True)
```

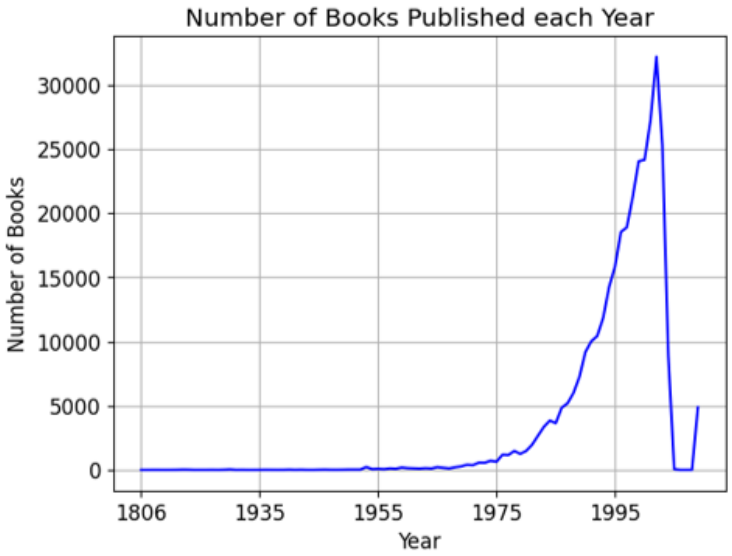
V Exploratory Data Analysis

1. Check for Correlation



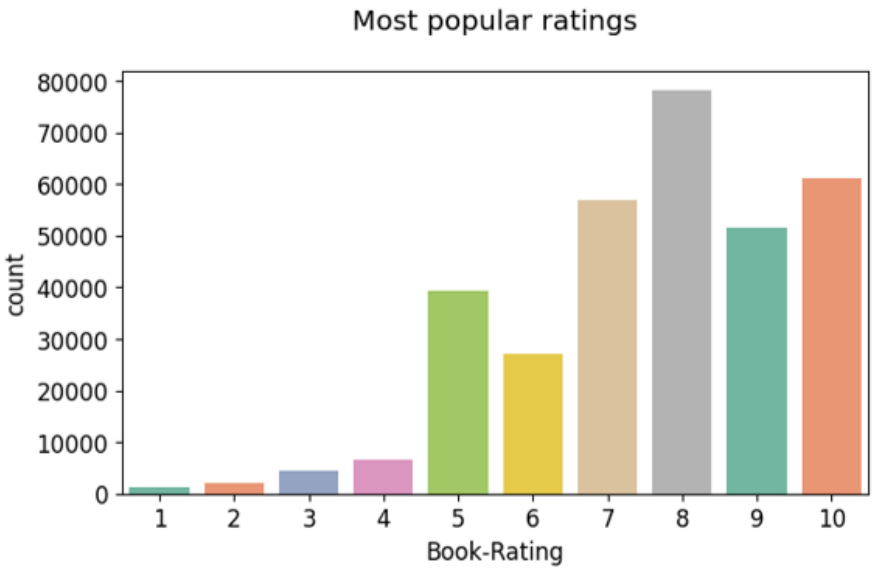
Clearly, from this plot, the three variables do not really have a relationship with each other at all.

2. Amazon Book's Trend



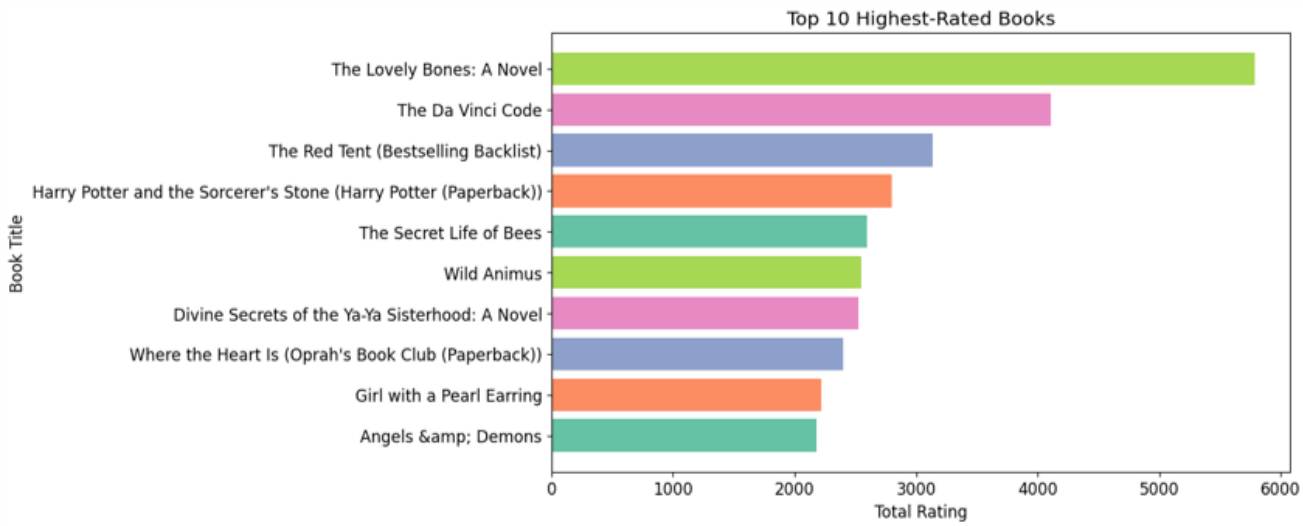
The increase of published books was due to the fact that Amazon was created in early 1990s. Then it went down around 2004-2007 due to bad economic conditions in the US.

3. Famous Ratings

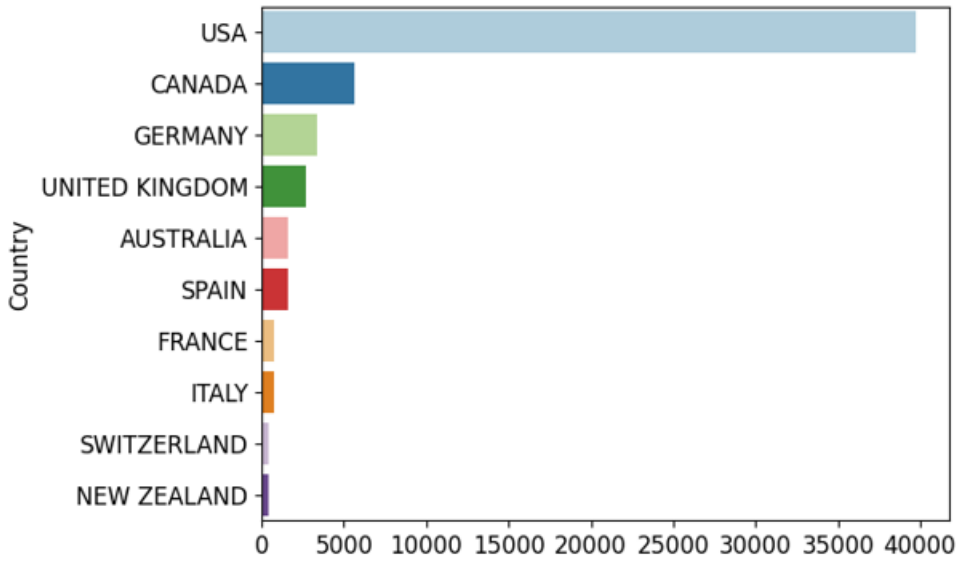


The most famous rating is 8. This indicates that users do not want to rate it as too good or too bad, meaning it is rated as neutral.

4. Famous Books

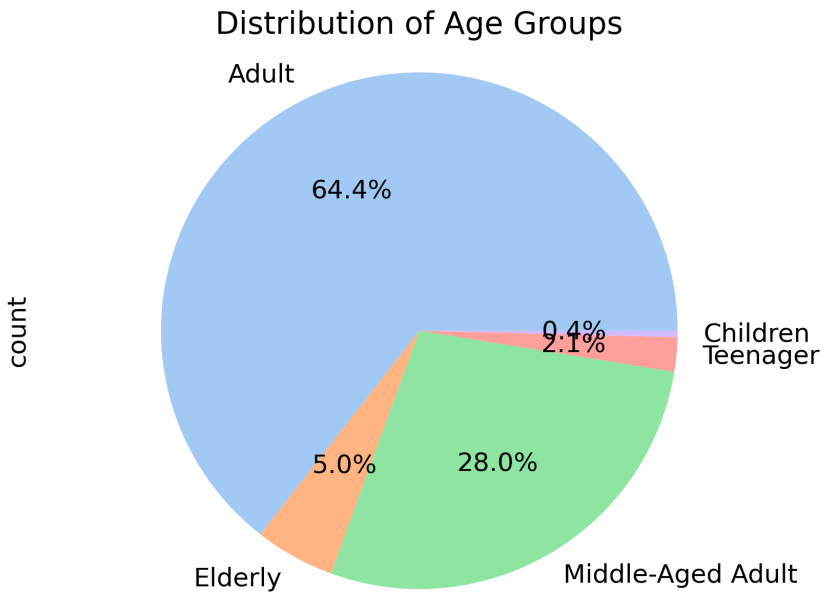


5. Countries with their Number of Users



USA is the country that has the most users since Amazon is originated in the US.

6. Age of Users



The users mostly are adults. Adult is the stage of learning.

VI Recommendation System

In this project, we will be using two approaches towards the recommendation system.

- **Popularity Based recommendation system :** is a system that recommends based on the popularity of certain things, for example: famous authors, famous books, etc.
- **Collaborative Filitering Based recommendation system :** When user A and user B read the same book, then user A read another book, then that book will be recommended to user B.

+ Types of Collaborative Filtering :

- **Model-based CF** uses machine learning algorithms to predict users' rating of unrated items. There are many model-based CF algorithms, the most commonly used are matrix factorization models such as to applying a SVD to reconstruct the rating matrix.
- **Memory-based CF** use user rating historical data to compute the similarity between users or items. The idea behind these methods is to define a similarity measure

between users or items, and find the most similar to recommend unseen items. There are two types of Memory-Based :

- **User-Based:** makes recommendations based on the user's preferences that are similar to other users. For example, if a user gives a similar rating to movies as the user in question. We could assume that they have similar interests. Thus, if the other user has seen and liked a movie that the user hasn't seen, we would recommend it.
- **Item-based:** suggests items similar to other items the active user liked. For example, if a user liked a Lord of the Rings book, then we would recommend another Lord of the Rings book.

6.1 Popularity Based Recommendation System

6.1.1 Famous Books

By counting the numbers of Ratings a certain book has, the top 10 books :

ISBN	Book-Rating	Book-Title	Book-Author	Year-Of-Publication	Publisher
0316666343	707	The Lovely Bones: A Novel	Alice Sebold	2002	Little, Brown
0971880107	581	Wild Animus	Rich Shapero	2004	Too Far
0385504209	487	The Da Vinci Code	Dan Brown	2003	Doubleday
0312195516	383	The Red Tent (Bestselling Backlist)	Anita Diamant	1998	Picador USA
0060928336	320	Divine Secrets of the Ya-Ya Sisterhood: A Novel	Rebecca Wells	1997	Perennial
059035342X	313	Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))	J. K. Rowling	1999	Arthur A. Levine Books
0142001740	307	The Secret Life of Bees	Sue Monk Kidd	2003	Penguin Books
0446672211	295	Where the Heart Is (Oprah's Book Club (Paperback))	Billie Letts	1998	Warner Books
044023722X	281	A Painted House	John Grisham	2001	Dell Publishing Company
0452282152	278	Girl with a Pearl Earring	Tracy Chevalier	2001	Plume Books

6.1.2 Country

Similar to above, but recommend the famous books based on specific country.
The top 5 books in France :

ISBN	Book-Rating	Book-Title	Book-Author	Year-Of-Publication	Publisher
2290311782	18	Je Voudrais Que Quelqu'un M'Attende Quelque Part	Anna Gavalda	2001	Editions J'Ai Lu
2253044903	15	Le Parfum : Histoire d'un meurtrier	Patrick Süskind	1988	LGF
2253150711	15	Stupeur Et Tremblements	Amelie Nothomb	2001	Distribooks
2070408507	12	Le Petit Prince	Antoine de Saint-Exupéry	1999	Gallimard
2266104535	11	Et Si C'Etait Vrai / If This Were Only True	Marc Levy	2001	Pocket

6.1.3 Weighted Average Ratings

In this section we using the model weight average for each Book's Average Rating. From one of the most popular database website **IMDb** (an acronym for **Internet Movie**

Database) owned by amazon. The model can be shown below:

$$W = \frac{Rv + Cm}{v + m}$$

Where

W = Weighted Rating

R = Average of the Books rating

v = No of people who have rated the books(number of votes)

m = minimum no of votes to be listed

C = the mean rating across all the books

The reason behind this is we want to make sure that our program recommending the right book to the reader with reliability and quality. For example, we have two books named it as A and B, Book A has 3.5 rating, and another one is 3 rating. Let say Book A has 5 user rating and Book B has 100 user rating. This show that although Book A have more rating but we can't assume that it is more reliable than Book B due to Book B have more user rating.

However, there might be some questions to the other aspect of it. One of them is 'What about the book that has just published? It may be low on the quantity of rating, but it can reliable as well'. To answer that, we can simply say that yes it might be reliable however it is not in the current time until it rated and read my many people, because once again we can't assume that.

6.1.4 Famous Authors

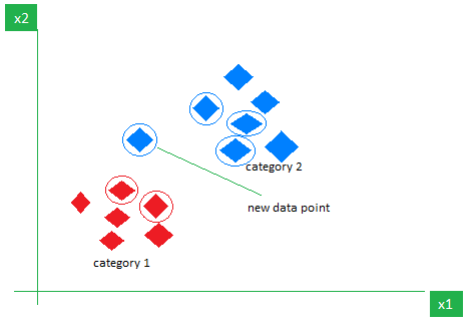
Let say you read Harry Potter and the Chamber of Secrets (Book 2)
Our recommendation output:

Book-Title	weighted_average
Harry Potter and the Chamber of Secrets Postcard Book	9.52
Harry Potter and the Goblet of Fire (Book 4)	9.21
Harry Potter and the Order of the Phoenix (Book 5)	9.01
Harry Potter and the Prisoner of Azkaban (Book 3)	8.99
Harry Potter and the Sorcerer's Stone (Book 1)	8.94

6.2 Collaborative Filitering Based Recommendation System

6.2.1 Memory Based (KNN)

KNN algorithm is an algorithm that finds the K nearest neighbors to a given data point based on a distance metric, such as Cosine Similarity.



Now, given another set of data points (also called testing data), allocate these points to a group by analyzing the training set. Note that the unclassified points are marked as 'White'. And in order to calculate the distance between the point to it **Nearest Neighbor** we can use the distance metrics.

+ Distance Metrics Used in KNN Algorithm

As we know that the KNN algorithm helps us identify the nearest points or the groups for a query point. But to determine the closest groups or the nearest points for a query point we need some metric. For this purpose, we use Cosine Similarity metric.

- **Cosine Similarity** is a measure of similarity between two non-zero vectors defined in an inner product space. Cosine similarity is the cosine of the angle between the vectors; that is, it is the dot product of the vectors divided by the product of their lengths.

- **Cosine Similarity** = $S_C(A, B) := \cos(\theta) = \dfrac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \dfrac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$

+ Implement KNN to our recommendation

First we have to filter out book that have rating below 3 and book that have number of rating frequency under 10% in our dataset.

```
# creating a pivot table
table = filter_df.pivot_table(columns='user_id', index='book_title', values=
 'book_rating')
table.fillna(0, inplace=True)
```

```
# converting to sparse matrix to save memory
from scipy.sparse import csr_matrix
sparse = csr_matrix(table)
```

```
#Creating an instance of KNN
from sklearn.neighbors import NearestNeighbors

model = NearestNeighbors(metric = 'cosine', algorithm='brute')
model.fit(sparse)
```

```
# function to get the recommended books
def recommender(book_name):
book_id = np.where(table.index == book_name)[0][0]
distance, suggestion = model.kneighbors(table.iloc[book_id, :].values.
 reshape(1, -1), n_neighbors = 6)

for i in range(len(suggestion)):
books = table.index[suggestion[i]]
for j in books:
print(j)
```

```
# Test the model
recommender("Harry Potter and the Goblet of Fire (Book 4)")
```

Output:

Harry Potter and the Goblet of Fire (Book 4)

Harry Potter and the Prisoner of Azkaban (Book 3)

Harry Potter and the Sorcerer's Stone (Book 1)

Harry Potter and the Order of the Phoenix (Book 5)

Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))

6.2.2 Model Based (SVD)

The Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into three matrices. For a given matrix A (of dimensions m x n), SVD represents it as the product of three matrices: $A = U \sum V^T$.
where

- U : m x m matrix of the orthonormal eigenvectors of AA^T.
- $\sum$: diagonal matrix with r elements equal to the root of the positive eigenvalues of AA^T or A^TA (both matrics have the same positive eigenvalues anyway).
- V^T : transpose of a n x n matrix containing the orthonormal eigenvectors of AA^T.

+ Fun Fact : Singular Value Decompositions (SVD) have become very popular in the field of Collaborative Filtering. The winning entry for the famed Netflix Prize had a number of SVD models.

```
#import the data
ratings_explicit = pd.read_csv('/content/explicit_rating.csv')
```

+ Surprise is a module of Python that is mainly focused on Recommendation System

```
# Convert the original dataset into the surprise type dataset, so that it
  would be compatible
from surprise import Reader, Dataset
reader = Reader(rating_scale = (1, 10))
data = Dataset.load_from_df(ratings_explicit, reader)
```

```
# import the model
from surprise import SVD, model_selection, accuracy
model = SVD()
```

```
# Split the data into train and test set with 80:20 ratio
trainset, testset = model_selection.train_test_split(data, test_size=0.2)
model.fit(trainset)
```

```
# Calculate the RMSE of the model
predictions = model.test(testset)
accuracy.rmse(predictions)
```

```
# Test the model
uid = 276729
iid = '0521795028'
```

```
pred = model.predict(uid, iid, verbose=True)

print(f'The estimated rating for the book with ISBN code {pred.iid} from
  user #{pred.uid} is {pred.est:.2f}\n')
actual_rate = ratings_explicit[(ratings_explicit.user_id == pred.uid) & (
  ratings_explicit.isbn == pred.iid)].book_rating.values[0]
print(f'The real rating given for this was {actual_rate:.2f}.')
```

Output : The estimated rating for the book with ISBN code 0521795028 from user #276729 is 6.60. The real rating given for this was 6.00.

+ Then we create some functions to display the top 10 books that we predict that the user rate high.

```
# Get all top 10 books
ex_readinglist = get_reading_list(userid)
for book, rating in ex_readinglist.items():
print(f'{book} : {rating} ')
```

Output :

The Perks of Being a Wallflower : 9.844579477651639

Two for the Dough : 9.712038898302719

The Girls' Guide to Hunting and Fishing : 9.442335858010466

Midnight in the Garden of Good and Evil : 9.33308819287904

Fast Food Nation: The Dark Side of the All-American Meal : 9.330570714743708

Look at Me : 9.304151799756394

Fried Green Tomatoes at the Whistle Stop Cafe : 9.173842418490702

Roots : 9.053343020845439

A Confederacy of Dunces (Evergreen Book) : 9.044892166917068

The Monster at the End of This Book : 9.027050615916968

VII Conclusions

In conclusion, the implementation of a book recommendation system offers numerous benefits to readers. By using advanced algorithms and personalized data analysis, the system provides tailored recommendations that match the unique preferences and reading history of each individual. This not only saves readers time and effort in searching for their next read but also introduces them to a diverse range of authors, and perspectives they may not have explored otherwise. Overall, the book recommendation system enhances

the reading journey and connects readers with books they are likely to love.

VIII References

- Wikipedia
- IMDB
- GeeksForGeeks - KNN
- GeeksForGeeks - SVD
- RadhikaRM from Github
- DSwithBappy