INSTITUTE FOR ADVANCED
COMPUTING AND
SOFTWARE DEVELOPMENT
AKURDI, PUNE

Documentation On

"Book Recommendation System"

PG-DBDA FEB 2020

Submitted By:

Group No:06

Gayatri Badgujar-1308

Priyanka Kothawade-1325

Mr.PrashantKarhale
CentreCoordinator

Mr. Akshay Tilekar
ProjectGuide

CERTIFICATE

This is to certify that the Project Entitled

Book Recommendation System

Submitted by:

Gayatri Badgujar-1308

Priyanka Kothawade-1325

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of Mr. AKSHAY TILEKAR and it is approved for the partial fulfilment of the requirement of Post Graduate Diploma in Big Data Analytics.

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the preliminary project report on “BOOK RECOMMENDATION SYSTEM”

I would like to take his opportunity to thank my internal guide Mr. AKSHAY TILEKAR for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to Mr. PRASHANT KARHALE, Centre Coordinator, Akurdi , Pune for his indispensable support, suggestions. In the end our special thanks to IACSD for providing various resources such as laboratory with all needed software platforms, continuous Internet connection, for our Project.

Gayatri Badgujar

Priyanka Kothawade

INDEX
TABLE OF CONTENTS

LIST OF FIGURES AND TABLES

Fig 7- Histogram of average rating..... 16
Fig 8- Content Based Recommendation Author wise 17

ABSTRACT

Recommendation Systems are one powerful tool that is at the heart of several successful companies such as Amazon, Netflix, Spotify and many others. They are designed to predict user's ratings or preference for products or items, based on criteria such as which items users have searched for, or rated favourably in the past, or what songs or movies they have liked. The goal is to expose users to new items with which they are more likely to engage or buy. The most advanced current recommendation systems in use today apply an ensemble of several different methods to give the most accurate results, including Natural Language Processing and Convolutional Neural Networks, but in this project, we made simple Content based and Collaborative Filtering recommendation system by applying a great library called surprise. Based on the user's and item's information available, these techniques provide recommendation's to users in their area of interest. Recommender systems have wide applications like providing suggestive list of items to customers for online shopping, recommending articles or books for online reading, movie or music recommendations, news recommendation etc.

CHAPTER 1
INTRODUCTION

In simple words, a recommendation system is any system that automatically suggests content for website readers and users. These systems were evolved as intelligent algorithms, which can generate results in the form of recommendations to users. [1] They require a large dataset and a fast-computing system that can perform analytics on same within fraction of seconds. [2] A variety of techniques have been proposed till today for performing recommendations. Recommendation system is defined as the computer program that helps the user determine goods and content by predict the users rating of each item and presentation them the substance that they would rate highly. There are different ways to approach a recommendation system. They are basically divided into Content-based and Collaborative Filtering systems.

1.1 Content Based System

Content-based systems are built from information on the characteristics of the items themselves. It's what you see when you are offered "other items like this". They can be very useful because users are likely to enjoy another book or movie similar to the one they are searching for. One disadvantage of this type of system is that some manual or semi-manual tagging is necessary.

1.2 Collaborative Filtering

The collaborative filtering is based on customer's behaviour's and past likes. It is used to make automatic predictions, filtering collecting preferences, interest of a user and taste information of many users. Collaborative filtering can be performed in two ways mainly, model-based and memory-based.

1.1 Purpose

The objective of the problem is to recommend users books based on their interest and previous records. Clearly, we have to create a recommendation system here. We have to use content based and collaborative filtering methods using KNN algorithms. We need to create a model that is accurate.

1.2 Scope of the project

Initial functional requirement will be: -

- Extracting Features.
- Choosing the optimum algorithm from set of algorithms.
- Testing.
- Analysis of result.

1.2 Software Life Cycle Model:

In order to make this Project we are going to use Classic LIFE CYCLE MODEL. Classic life cycle model is also known as WATER FALL MODEL. The life cycle model demands a Systematic sequential approach to software development that begins at the system level and progress through analysis design coding, testing and maintenance.

Fig 1. Software life cycle model

1.2.1 The Classic Life Cycle Model

The waterfall model is sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception initiation, Analysis, Design (validation), construction. Testing and maintained.

System Engineering and Analysis:
Because software is always a part of larger system work. Begins by establishing requirement for all system elements and Then allocating some subset of these requirement to the software system Engineering and analysis encompasses the requirement gathering at the system level with a small amount of top level design and analysis.

Software requirement Analysis:
The requirement gathering process is intensified and focused specifically on the software Requirement for the both system and software are discussed and reviewed with the customer. The customer specifies the entire requirement or the software and the function to be performed by the software.

Design:
Software design is actually a multi-step process that focuses on distinct attributes of the program data structure, software architecture, procedural detail and interface of the software that can be assessed or quality before coding begins. Like requirement the design is documented and becomes part of the software.

Coding:
The design must be translated into a machine-readable form. The coding step performs this task. If design is programmed in a detailed manner, coding can be accomplished mechanically.

Testing:
Once code has been generated programmed testing begins. The testing process focuses on the internals of the software ensuring that all statement have been tested and on the functional externals hat is conducting tests to uncover the errors and ensure that defined input will produce the results that agree with the required results.

Unit testing:
In computer programming, Unit testing is software Verification and validation method where the programmer gains confidence that individual units of source code are fit to use A unit is the smallest testable part of an application. In procedural programming a unit may be an individual programmed, function, procedure, etc. while in object-oriented programming, the smallest Unit is a class, which may belong to a base/super class abstract class or derived/child class.

Benefits:
The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict written contract that the piece of code must satisfy.

Documentation:
Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit and how to use it can look at the tests to gain a basic understanding of the unit API.

Limitation of unit testing:

Testing cannot be expected to catch error in the program –It is impossible to evaluate all execution paths for all but the most trivial programs. The same is true for unit testing. Additionally, by unit testing only types the functionality if the units themselves.

Maintenance:

Software will undoubtedly undergo change after it is Delivered to the customer. Change will occur because errors have been encountered because the software must be able adopted to accommodate changes in its external environment because the customer requires functional or performance enhancement enhancements. The classic life cycle is the oldest and most widely used paradigm or software engineering.

CHAPTER 2
DATASET DESCRIPTION

In this project we used Books dataset and ratings dataset. We found these datasets from Kaggle.

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 26 columns. We dropped unnecessary columns from the merged dataset. And rename some column names. There were some NA values which we dropped those NA values and after cleaning of the dataset the final Book data dataset consists of 16 columns.

As many authors writing same books many years. In this case all columns have the same value for such 'duplicate' books except for the year. Especially the number of reviews and the user rating represent the total number overall years. So, for that we work with a data frame of unique book data unless otherwise stated. In case of duplicates the book is assigned to the first year it has been a displayed. By using 'last', the last occurrence of each set of duplicated values is set on False and all others on True. By setting keep on False, all duplicates are True. To find duplicates on specific column(s), use subset. And finally, we have unique book dataset which consists of 784 rows and 16 columns.

CHAPTER 3
METHODOLOGY

3.1 Flow diagram

Figure 1 displays the Book Recommendation Model flow graph method. This chart shows the flow of the planned program operation. Method flow demonstrates how well the system works, how much the system handles the relevant data, and how ranking is expected by the system.

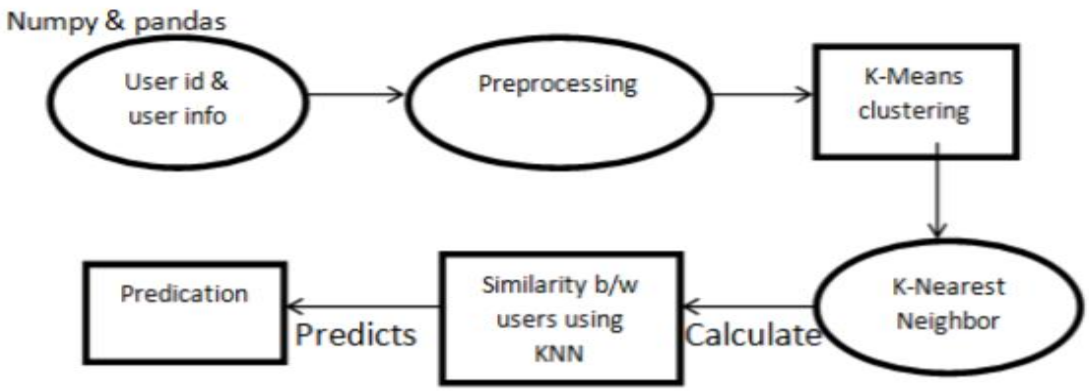


Fig 2. Flowchart of the system

We got the dataset from Kaggle and it has three data files books and ratings. Using the NumPy, pandas, sklearn and Matplotlib library the separate the raw data frames and processing.

Squared techniques are used to find the correct number of clusters so that K-means clustering can be functional to the books.

After applying K-means clustering a clustered matrix factorization is build which defined average rating the user gives to each cluster. Pearson correlation similarity between the users is calculated. Finally find the accuracy through root means square error.

CHAPTER 4
ALGORITHM DESCRIPTION

4.1 Content Based Recommender

A content-based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link). Based on that data, a user profile is generated, which is then used to make suggestions to the user. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate. The concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) are used in information retrieval and also content-based filtering mechanisms (such as a content-based recommender). They are used to determine the relative importance of a document / article / news item / movie etc.

4.1.1 Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF is in the sub-area of Natural Language Processing (NLP). It is used in information retrieval for feature extraction purposes. In short, you are somehow counting the occurrence of each words in a document and weight the importance of each words, and calculate a score for that document.

Term Frequency

Frequency of the word in the current document to the total number of words in the document. It signifies the occurrence of the word in a document and gives higher weight when the frequency is more so it is divided by document length to normalize.

$$\mathrm{Tf}(t) = \frac{\text{Frequency occurence of term t in document}}{\text{Total number of terms in document}}$$

Inverse Document Frequency

Total Number of Documents to the frequency occurrence of documents containing the word. It signifies the rarity of the word as the word occurring the document is less the IDF increases. It helps in giving a higher score to rare terms in the documents.

$$\mathrm{Idf}(t) = log_{10}(\frac{\text{Total Number of documents}}{\text{Number of documents containing term t}})$$

In the End, TF-IDF is a measure used to evaluate how important a word is to a document in a document corpus. The importance of the word increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

TF-IDF Vector

After defining the TF-IDF value for the tags, then we can make the keyword vectors for each item. Each row below represents a keyword vector for one item.

Compare the Similarity of the item TF-IDF vector

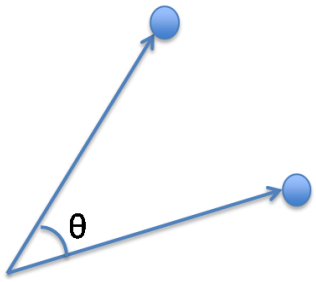
To compute how similar the item vectors are, we will can use various methods such as:

- Cosine Similarity
- Euclidean Distance
- Peason's Correlation

Cosine Similarity

Cosine similarity is a measure of similarity between two non-zero vectors. By Vector representation we can see how closely related two sentences 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.

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$



4.2 Collaborative Filtering

Collaborative Filtering systems are based on users' ratings for items. They are calculated by using a collection of user's ratings of items. The idea behind it is that similar users will have similar preferences and that users have a tendency to like items that are similar to each another. This type of recommendation system can use two different approaches to calculate this similarity.

The collaborative filtering is based on customer's behaviors' and past likes. It os used to make automatic predictions, filtering collecting preferences, interest of a user and taste information of many users. Collaborative filtering can be performed in two ways mainly, model-based and memory-based.

4.2.1 Memory-based:

User based Collaborative Filtering

In order to make a new recommendation to a user, user-user method roughly tries to identify users with the most similar "interactions profile" (nearest neighbors) in order to suggest items that are the most popular among these neighbors (and that are "new" to our user). This method is said to be "user-centered" as it represents users based on their interactions with items and evaluate distances between users.

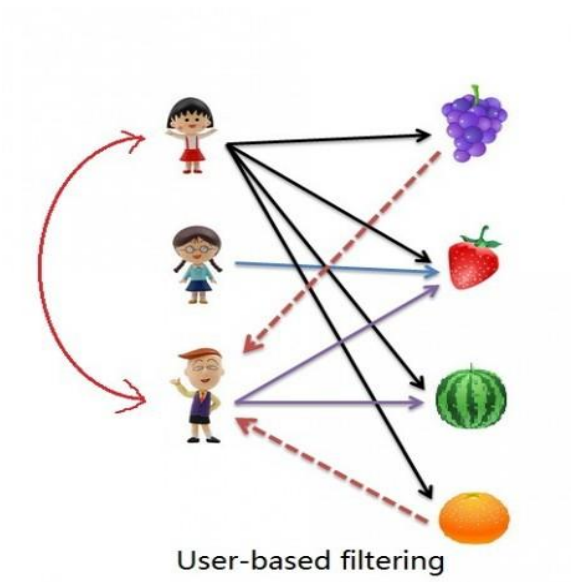


Fig 3: User based Collaborative Filtering

Item based Collaborative Filtering

To make a new recommendation to a user, the idea of item-item method is to find items similar to the ones the user already "positively" interacted with. Two items are considered to be similar if most of the users that have interacted with both of them did it in a similar way. This method is said to be "item-centered" as it represents items based on interactions users had with them and evaluate distances between those items.

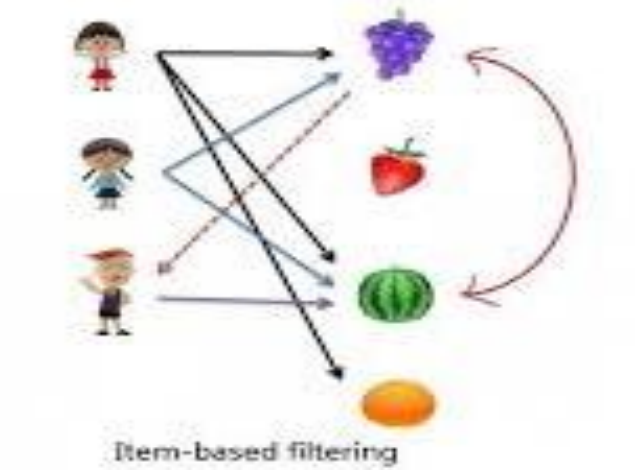


Fig 4 : Item based Collaborative Filtering

CHAPTER 5

MODELS

Implementing Knn

We convert our table to a 2D matrix, and fill the missing values with zeros (since we will calculate distances between rating vectors). We then transform the values(ratings) of the matrix dataframe into a SciPy sparse matrix for more efficient calculations.

Collaborative Filtering using k-Nearest Neighbors (kNN)

The standard method of Collaborative Filtering is known as Nearest Neighborhood algorithm. There are user-based CF and item-based CF. Let's first look at User-based CF. We have an n × m matrix of ratings, with user u_i, i = 1, …n and item p_j, j=1, …m. Now we want to predict the rating r_{ij} if target user i did not watch/rate an item j. The process is to calculate the similarities between target user i and all other users, select the top X similar users, and take the weighted average of ratings from these X users with similarities as weights.

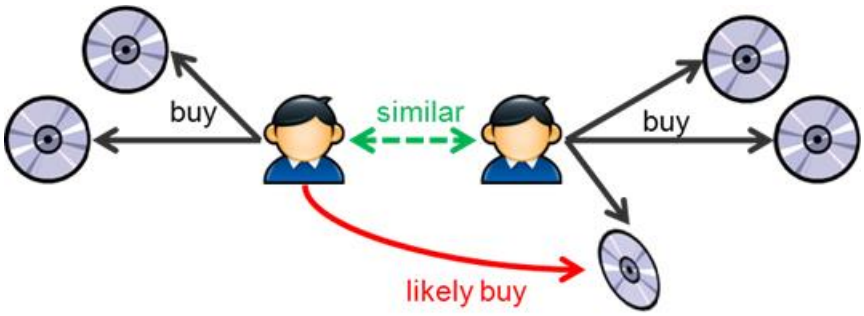


Fig 5 Nearest Neighbor user based-collaborative filtering

Pivot Matrix

- It converts the ratings table to a 2D matrix
- We can get the combination of user id title and rating index as user id
- NaN values are which whether the person give the ratings to the books or not

```
userRatings = train.pivot_table(index=['user_id'],columns=['title'],values='rating')
userRatings.head()
```

title	'Salem's Lot	'Tis (Frank McCourt, #2)	1421: The Year China Discovered America	1776	1984	A Bend in the River	A Bend in the Road	A Brief History of Time	A Briefer History of Time	A Case of Need	...	Wolves of the Calla (The Dark Tower, #5)	Women in Love (Brangwen Family, #2)	World War Z: An Oral History of the Zombie War	World Without End (The Kingsbridge Series, #2)	Wuthering Heights
user_id																
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN

Fig. 6 Pivot Matrix

After fitting that matrix we will use kneighbors for to find out which books/products are Similar. In this By Applying Collaborative filtering using K-Nearest Neighbors, we will convert that 2D Matrix into and fill missing values with zeroes others with given ratings, Now we will calculate distances between rating vectors, then we will transform the values(ratings) of matrix dataframe into a SciPy sparse for more efficient calculations.

After fitting & applying NearestNeighbors ->

This will find the distances/closeness of instances. It then classifies an instance by finding the Nearest Neighbors and picks the most popular & closer class.

CHAPTER 6
RESULTS

Python libraries import: NumPy, Pandas, Matplotlib, and sklearn. The dataset contains three excel files. The dataset of 90,000 users includes 1.1 million reviews of 270,000 books. The scores are on (1-5) scale.

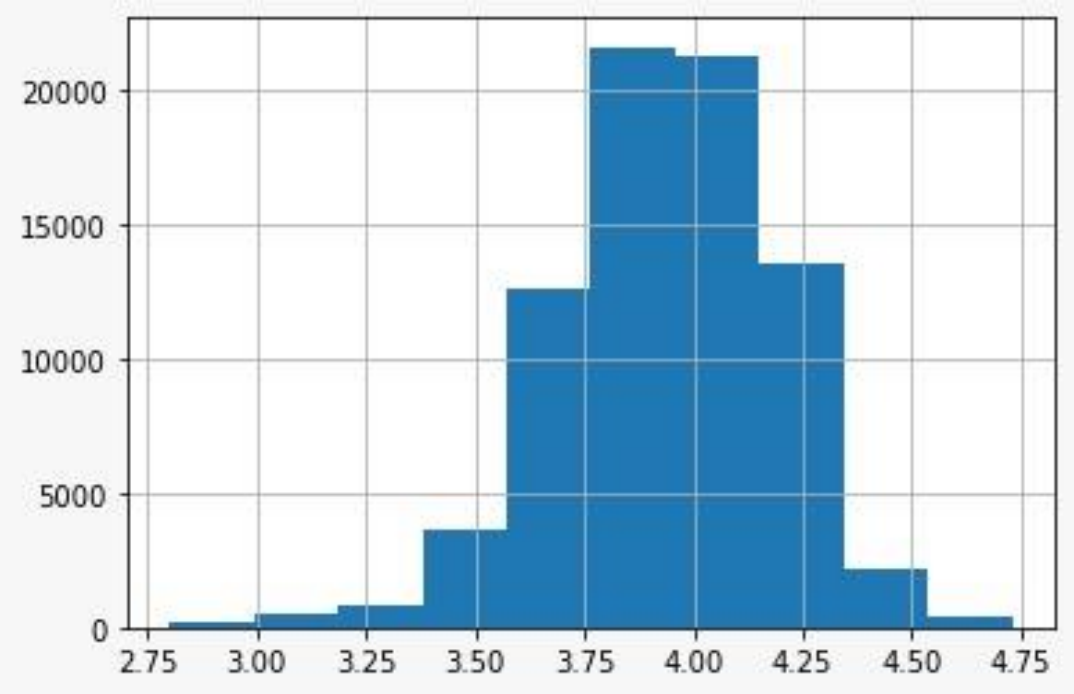


Fig 7: Histogram Distribution of Average user rating

From above visualization we can see that average rating is mostly between 3.5 to 4.5

Content Based Recommendation Author Wise

```
In [96]: 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)
```

```
In [97]: 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)
```

Fig 8. Content Based Recommendation Author Wise

Item-based Collaborative Filtering

```
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
```

Fig. 9 Item-based Collaborative Filtering

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
```

Fig. 10 User-based Collaborative Filtering

CHAPTER 7

ACCURACY

To calculate the accuracy of our predictions I available to use an ordinary statistical metric named root mean square error (RMSQ). RMSE is a measurement of the variation between the user's real books ratings and the We predicted for the same books. If the lower the RMSE, the more acceptable the model. An RMSE of zero means our model is absolutely guess the user ratings

User Based Collaborative Filtering:

User-Based Collaborative Filtering is a technique used to predict the items that a user might like on the basis of ratings given to that item by the other users who have similar taste with that of the target user. Many websites use collaborative filtering for building their recommendation system.

```
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

0.9851997664199224
```

Fig 11: User based Accuracy

As we can see that the RMSE score is 0.98 and Model 0.9851997664199224

Item Based Collaborative Filtering :

Item-based: For an item I, with a set of similar items determined based on rating vectors consisting of received user ratings, the rating by a user U, who has rated it, is found by picking out N items from the similarity list that have been rated by U and calculating the rating based on these N ratings.

```
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
```

RMSE score is 0.9549 and Item-based Knn Model Accuracy is 95.49235500386906

Fig 12: Item based Accuracy

CHAPTER 8
CONCLUSIONS

To conclude about our project, we have made analysis of different research papers and algorithm implemented in it about recommendation systems. 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 covers dataset and ratings dataset and KNN algorithm to display most similar books based on User based and Item based Collaborative Filtering.

CHAPTER 9
REFERENCES

[1] Ahuja, R. (2019). Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbour. IEEE.

[2] Dara, S. (2019). A survey on group recommender systems. Journal of Intelligent Information Systems.

[3] Gao, Y. (2019). Research on Book Personalized Recommendation Method Based on Collaborative Filtering Algorithm. IEEE.

[4] M, I. (2020). Predicting Books' Overall Rating Using Artificial Neural Network. International Research Journal of Engineering and Technology.

[5] Parvatikar, S. (2015). Online Book Recommendation System by using Collaborative filtering and Association Mining. IEEE.

[6] Ramakrishnan, G. (2020). Collaborative filtering for book recommendation system. SocProS.