RECOMMENDATION SYSTEM

A Project Report

Submitted in partial fulfillment of the

Requirements for the award of the degree

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Miss. Rutika Ulhas Warge

FI-12

Under the esteemed guidance of

Mr. Sachin Bhostekar

Lecturer

DEPARTMENT OF INFORMATION TECHNOLOGY

JANATA SHIKSHAN MANDAL**'s**

Smt. Indirabai G. Kulkarni Arts, J. B. Sawant Science College & Sau.

Jankibai Dhondo Kunte Commerce College

(Affiliated to University of Mumbai)

ALIBAUG, 402 201

MAHARASHTRA

AY 2021-22

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the project entitled, "Recommendation System"**, is bonafied work of** Mr. Sachin Bostekar **bearing Seat.No: ____ submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.**

Internal Guide Coordinator

External Examiner

Date: College Seal

ABSTRACT

Recommendation Systems (RS) are commonly used in many Internet operations and their significance is growing as a result of the Internet's "Information Overload" crisis. The recommended framework will be reviewed in the first part. I will explain the three major types of suggested systems in the second section: collaborative, content-based and hybrid. I'll introduce a collaborative approach in the third segment. Based on the user's previous actions or explicit input, this approach uses item features to suggest other products that are close to what they want. I' am aiming to go along with e-commerce as a domain. Buying and selling items is simple on e-commerce platforms. As a result, we've chosen an example as product data. I will recommend the best product for the client based on their requirements after analysing data. In the fourth section, I will demonstrate how to use Python to implement collaborative Filtering to provide personalized recommendations to users

ACKNOWLEDGEMENT

The completion of this project would have been impossible without the guidance and support from my internal guide. Thus, I would like to thank those who gave me right instruction as to go about project.

Dedication, hard work, patience and correct guidance are required thing during project completion. Able and timely guidance not only help in making an effort fruitful but also transform the people of learning and implementation into on enjoyable experience.

In particular, I like to thank for the blessing to our **Principle Dr. A. K. Patil** who have always been source of inspiration. I wish to thanks **Prof Satyajeet Tulpule** (In-charge of Information Technology Dept.) for having faith in this project idea and granted support in all direction.

I'm grateful to our Project Coordinator **Mr. Sachin Bhostekar** for the internal guidance, inspiration and constructive suggestions that helpful us in the preparation of this project.

I would like to thank our college library and all my friends for their help in completion of my project reports.

I am grateful and thankful to all my teachers:

Miss. Dikshita Bhoir
Miss. Chaitali Chaudhari
Mr. Kundan Sawant

Who share their year of experience, excellent support and blossoms of suggestions with me.

DECLARATION

I hereby declare that the project entitled, "**Recommendation System**" done at place where the project is done, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

(Miss. Rutika Ulhas Warge)

Table of Contents

CHAPTER 1: Introduction 7
1.1: Background 7
1.2: Objective 7
1.3: Purpose 7
1.4: Applicability 8
1.5: Organization of Report 8
CHAPTER 2: Survey of Technologies 9
2.1: Types of Recommendation System 10
2.1.1: Collaborative Method 10
2.1.2: Content Based Method 10
2.1.3: Hybrid Method 11
CHAPTER 3: Requirements and Analysis 13
3.1: Problem definition 13
3.2: Requirements Specification 13
3.3 Planning and Scheduling 14
3.4: Domain 16
3.4.1: E-commerce 16
CHAPTER 4: System Design 17
4.1: Basic Module 17
4.2: Class diagram 18
..... 18
4.3: Use case diagram 19
CHAPTER 5: Implementation Approaches 20
5.1: Collaborative Approach 20
5.2: Types 20
5.2.1: Item-Item Relationship 21
5.3: Algorithm 22
5.4: Dataset 23
5.4.1: Attribute information 23

CHAPTER 6: RESULTS AND DISCUSSION 24
6.1: Test Report 24
6.2: User Documentation 25
CHAPTER 7: Conclusions 38
7.1: Conclusion: 38
7.1.1: Significance of the system 39
7.2: Limitation of the system 39
7.3: Future scope of the project 40
Bibliography 41

CHAPTER 1: Introduction

1.1: Background

Recommendation systems are machine learning systems that help users discover new product and services. Historically, the first recommendation system was the Tapestry which coming out in 1992 from Xerox PARC, then a variety of techniques and technologies of recommendation system have been produced and introduced.

1.2: Objective

The objective of recommender systems is **to provide recommendations based on recorded information on the users' preferences**. These systems use information filtering techniques to process information and provide the user with potentially more relevant items.

1.3: Purpose

Recommender systems are so commonplace now that many of us use them without even knowing it. Because we can't possibly look through all the products or content on a website, a recommendation system plays an important role in helping us have a better user experience. So, I'm interested to knowing that how recommendation system exactly work. In the later chapters we will explain the three major types of suggested systems. I'm using collaborative method for that I will introduce a collaborative approach. My further aim to go along with e-commerce as a domain. Next, I will demonstrate how to use python to implement collaborative filtering. The important component of any of these systems is, it takes information about the user and predicts the product to that user.

1.4: Applicability

- It is easy for a user to interpret the results.
- As only the highly rated items gets displayed, it becomes easy to implement.
- Data collection is relatively easier in this technique due to limited number of variables required.

1.5: Organization of Report

The basic structure of this project report follows the introduction of the system, the survey of technology, requirement analysis, and system design. However, rather than dive instantly into these topics, I have added this overview section so that you can come to a wide picture of the discipline. The chapters in this part are:

- Chapter 2 introduces available technology in the computer world. Survey and comparative studies of technology used for project development.
- Chapter 3 discusses functional and non-functional requirements of the systems. Description of planning and scheduling of project using PERT and GANTT chart. Discussing Hardware and software requirements for project execution and conceptual modeling.
- Chapter 4 continues with modularization of the system, data designing with schema, creating the data structure, creating relational schemas, designing user interface, applying security mechanism and finally examine the result for accuracy.

CHAPTER 2: Survey of Technologies

In this chapter, the information about the software required to complete the project is given. Details of other technologies used are also mentioned in this chapter. Recently many software and technologies are available to develop projects.

Popular Object-Oriented Languages are Java, Python, C++, Visual Basic .NET, C#, and Ruby is the most popular OOP languages today. According to DB-Engines, the most popular database systems are Oracle, MySQL, Microsoft SQL Server, PostgreSQL and MongoDB (db-engines, 2018). Some of UML modelling software are Microsoft Visio, Rational Rose, Star UML, Visual Paradigm, Magic Draw and much more available today.

I have been doing comparative study of the above software, technology and tools as follows.

1. Language and Frameworks:

Python: Python is a language that uses a simpler syntax than PHP. It' designed to have a very readable code, and for that reason is very recommended to learn programming. Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

- Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive − You can actually sit at a Python prompt and interact with the interpreter directly to write your problems.
- Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games. (Tutorial Point, 2018)

2.1: Types of Recommendation System

2.1.1: Collaborative Method

Collaborative filtering recommender makes suggestions based on how users rated in the past and not based on the product themselves. It only knows how other customers rated the product. For example, suppose user A assigned the rating to product X and Y and new user B who has assigned the same rating to product X but, hasn't purchase product Y yet. So, collaborative method will recommend him the product Y. Then, the main idea that rules collaborative methods is that these past user-item interactions are sufficient to detect similar users and similar items and make predictions based on these estimated proximities. (towardsdatascience, n.d.)

The main advantage of collaborative method is that, there is no requirement for product descriptions. so, they can be used in many situations. Moreover, the more users interact with items the more new recommendations become accurate: for a fixed set of users and items, new interactions recorded over time bring new information and make the system more and more effective.

However, as it only considers past interactions to make recommendations, collaborative filtering suffer from the "cold start problem": it is impossible to recommend anything to new users or to recommend a new item to any users and many users or items have too few interactions to be efficiently handled. This drawback can be addressed in different way: recommending random items to new users or new items to random users (random strategy), recommending popular items to new users or new items to most active users (maximum expectation strategy), recommending a set of various items to new users or a new item to a set of various users (exploratory strategy) or, finally, using a non-collaborative method for the early life of the user or the item.

2.1.2: Content Based Method

It is based on the information on the contents of the item rather than on the user opinions. The main idea is if the user likes an item, then he or she will like the "other"

similar item. Content based approaches use additional information about users and/or items. If we consider the example of a movies recommender system, this additional information can be, for example, the age, the sex, the job or any other personal information for users as well as the category, the main actors, the duration or other characteristics for the movies (items). (wikipedia, n.d.)

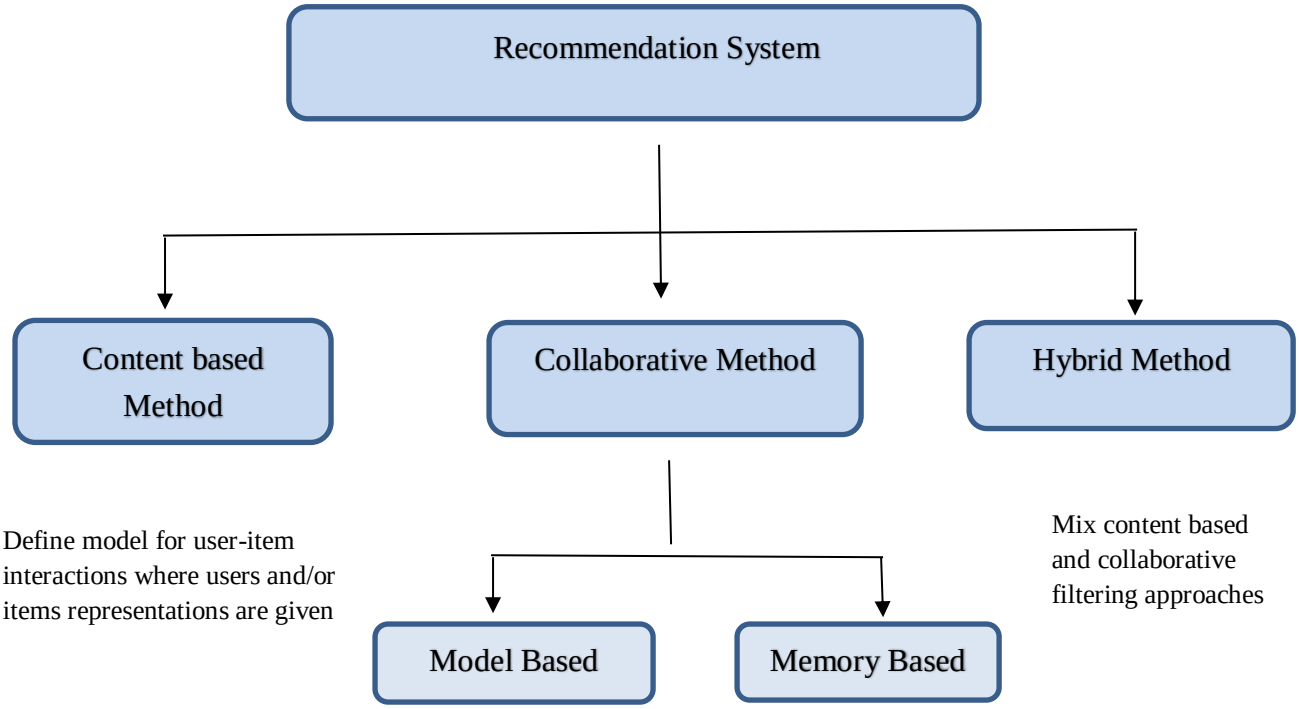
Then, the idea of content-based methods is to try to build a model, based on the available "features", that explain the observed user-item interactions. For example, to model the fact that young women tend to rate better some movies, that young men tend to rate better some other movies and so on. If we manage to get such model, then making new predictions for a user is pretty easy: we just need to look at the profile (age, genders, . . .) of this user and, based on this information, to determine relevant movies to suggest. Content based methods suffer far less from the cold start problem than collaborative approaches: new users or items can be described by their characteristics (content) and so relevant suggestions can be done for these new entities. Only new users or items with previously unseen features will logically suffer from this drawback, but once the system old enough, this has few to no chance to happen.

2.1.3: Hybrid Method

Recent research has demonstrated that a hybrid approach combining collaborative filtering and content-based filtering. Hybrid approaches can be implemented in several ways: by making content-based and collaborative-based predictions separately and then combining them; by adding content-based capabilities to a collaborative-based approach (and vice versa).

Examples of systems mixed content based method and collaborative method : Tapestry Fab, it suggests relevant URLs to users by combining users "ratings and Web pages" similarities and Quickstep, it supports Web page recommendation.

Netflix is a good example of the use of hybrid recommender systems. Because, when a new user subscribes to their service they are required to rate content already seen or rate particular genres. Once the user begins using the service, collaborative filtering is used and similar content is suggested to the customer.


Recommendation System
Content based Method
Collaborative Method
Hybrid Method
Define model for user-item interactions where users and/or items representations are given
Mix content based and collaborative filtering approaches
Model Based
Memory Based
Define a model for user-item interactions where users and terms representation have to be learned from interaction matrix
Define no model for user-item interaction and rely on similarities between users or item in terms of observed interactions

CHAPTER 3: Requirements and Analysis

3.1: Problem definition

Recommendation systems are software tools or knowledge discovery techniques that provide suggestions for items to a user. The aim of these systems is to recommend items that a user is likely to be interested in and learn more about user preferences and constraints. This project is based on improving the existing recommendation systems

3.2: Requirements Specification

The Requirement Analysis is a technology engineering software that has several task that determine the requirements or conditions to be fulfilled for new or changing product, considering the different needs of different users.

Functional requirement:

This section encapsulates the major software functions and data flow among the participants of the system. The participants include the user and the interagent. The user, in other words the end user.

Non- Functional requirement:

- Performance:

  Performance of making recommendation and updating this recommendation is very important issue because we are aiming to make the system real-timed. In other words, the system should have enough speed that users of the system cannot realize the processing of data. Besides, our web service should handle multiple users at the same time.

- Security:

Database has to be reached securely and its data should not be broken. It also should not change except interagent updates. Moreover, since our dataset contain some personal information of user such as user id, tracks he/she listened, security design is important in the web service.

- Usability:

The scope of the product is widespread. Besides, people from every age shall easily use the system.

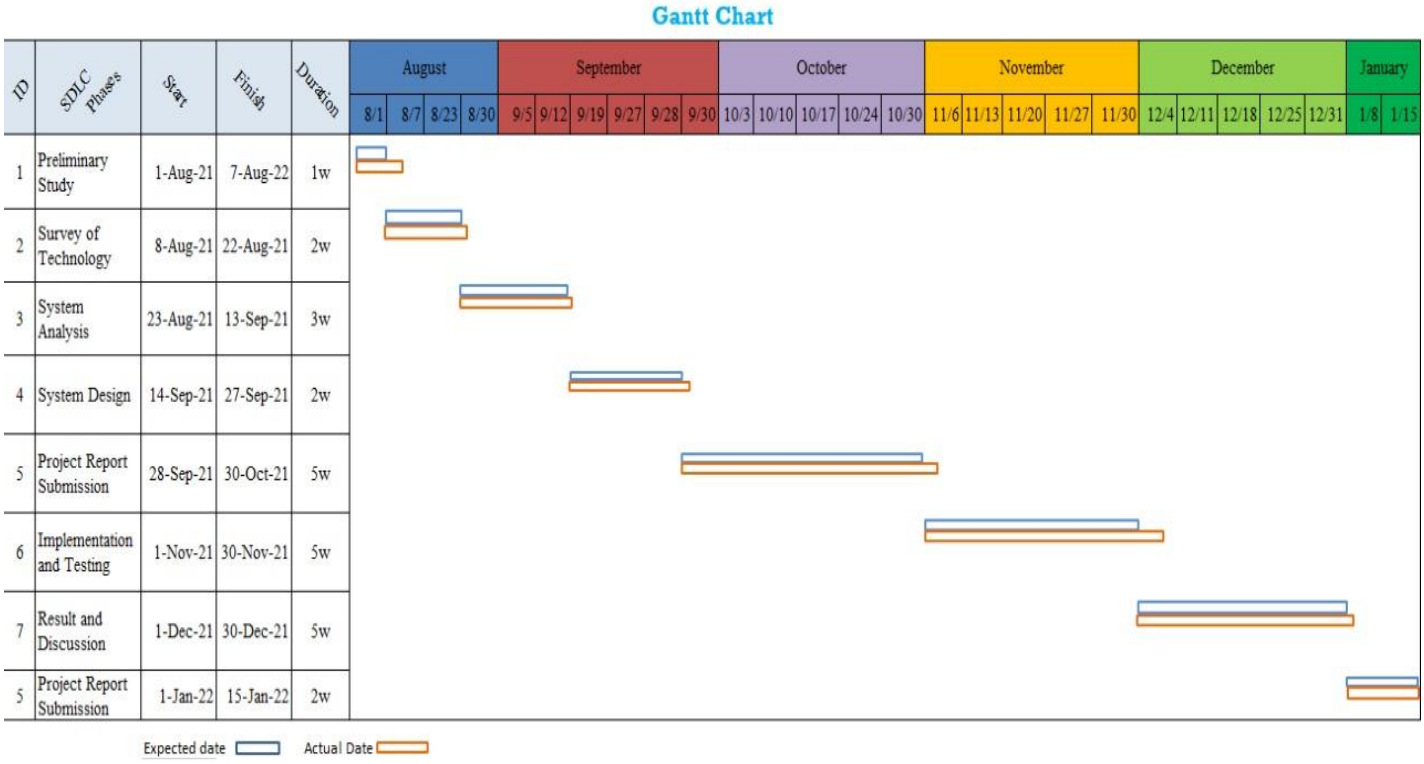
3.3 Planning and Scheduling

Project planning is one of the most important jobs of a software project. We have to break down the work into parts and assign these to project team members, anticipate problems that might arise, and prepare tentative solutions to those problems. The project plan, which is created at the start of a project, is used to communicate how the work will be done to the project team and customers, and to help assess progress on the project.

Project scheduling is the process of deciding how the work in a project will be organized as separate tasks, and when and how these tasks will be executed. We estimate the calendar time needed to complete each task, the effort required, and who will work on the tasks that have been identified. We also have to estimate the resources needed to complete each task, such as the disk space required on a server, the time required on specialized hardware, such as a simulator, and what the travel budget will be. In terms of the planning stages, an initial project schedule is usually created during the project start-up phase. This schedule is then refined and modified during development planning.

GANTT Chart:

Project schedules may simply be represented in a table or spreadsheet showing the tasks, effort, expected duration, and task dependencies. Also the graphical representations are made that are easier to read and understand. In this project, we use GANTT charts for representation. GANTT charts or Bar charts, which are calendar based, show who is responsible for each activity, the expected elapsed time, and when the activity is scheduled to being and end. Bar charts are sometimes called 'Gantt charts', after their inventor, Henry Gantt, the activity is scheduled to begin and end. Bar charts are sometimes called 'Gantt charts', after their inventor, Henry Gantt.



3.4: Domain

Recommended Systems have been widely used in many internet activities and it is worth mentioning some examples of the current actual uses of recommended system. However, we will mention e-commerce that use one or more methods of recommended system

3.4.1: E-commerce

The e-commerce becomes an important media to exchange products and services. Within e-commerce sites, buying and selling things between client and trader is easy, especially in our e-societies age. But the growth of e-commerce sites caused the product information overload; however, recommended system are used to solve this problem and are used “to suggest products to their customers and provide consumers with information to help them decide which products to purchase. There are more and more e-commerce businesses that use one or more variations of recommended system technologies in their Web sites”.

Examples: Amazon.com, Flipkart.com and CdNow.com.

CHAPTER 4: System Design

4.1: Basic Module

An Application Module is a logical container for coordinated objects related to a particular task, with optional programming logic. Application Modules provide a simple runtime data connection model (one connection per Application Module) and a context for defining and executing transactions.

There are mainly three modules in this propose project:

1. **NumPy:**

   NumPy, which stands for Numerical Python, is **a library consisting of multidimensional array objects and a collection of routines for processing those arrays**. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'. (mygreatlearning, n.d.)

2. **Pandas:**

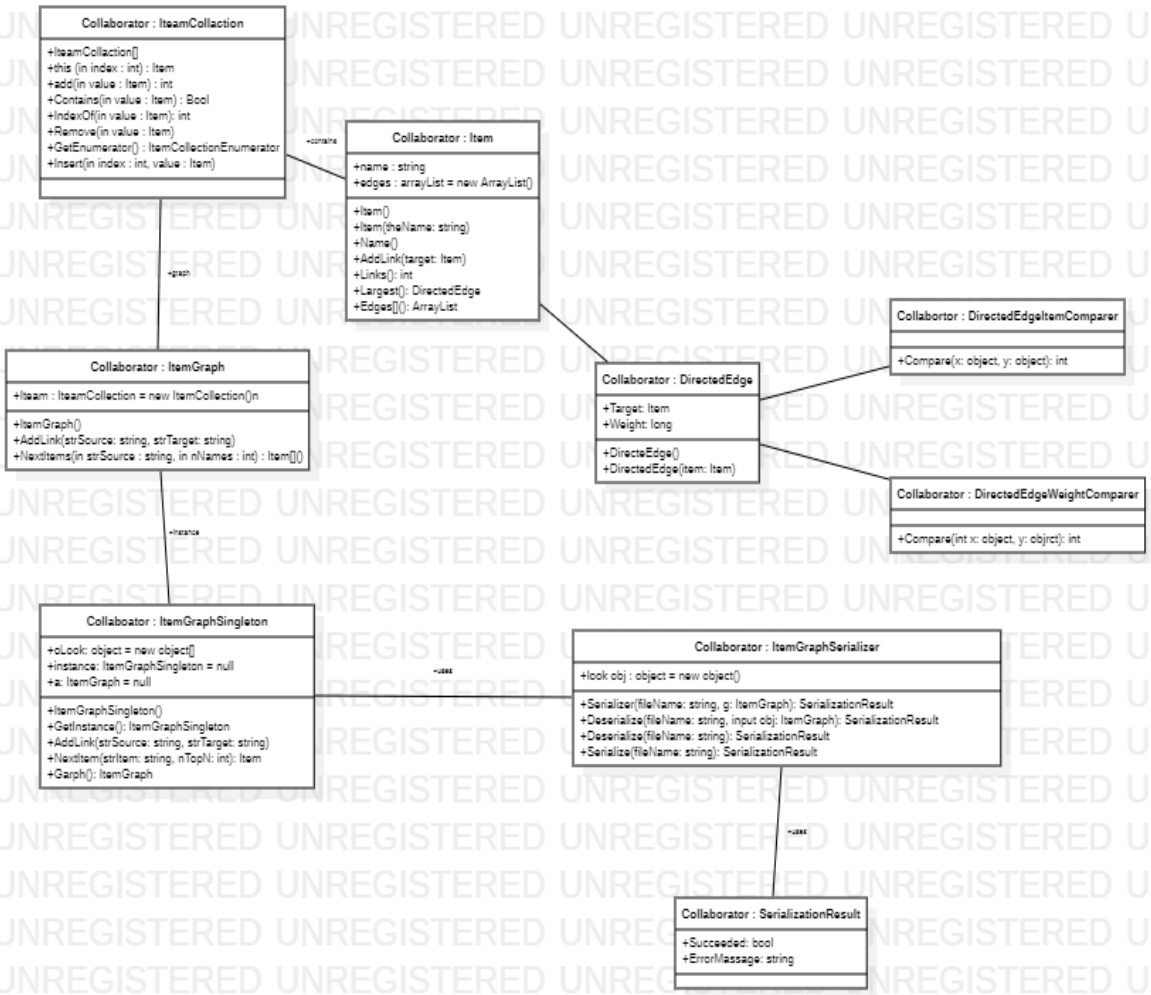
   Pandas is an open-source library in Python. It provides ready to use high-performance data structures and data analysis tools. NumPy is a low-level data structure that supports multi-dimensional arrays and a wide range of mathematical array operations. Pandas has a higher-level interface. It also provides streamlined alignment of tabular data and powerful time series functionality. Pandas provides a rich feature-set on the Data Frame. For example, data alignment, data statistics, slicing, grouping, merging, concatenating data, etc. (journaldev, n.d.)

3. **Matplotlib.pyplot:**

   Matplotlib is **a comprehensive library for creating static, animated, and interactive visualizations in Python**. Matplotlib makes easy things easy and hard things possible. Create publication quality plots. Make interactive figures that can zoom, pan, update. (matplotlib, n.d.)

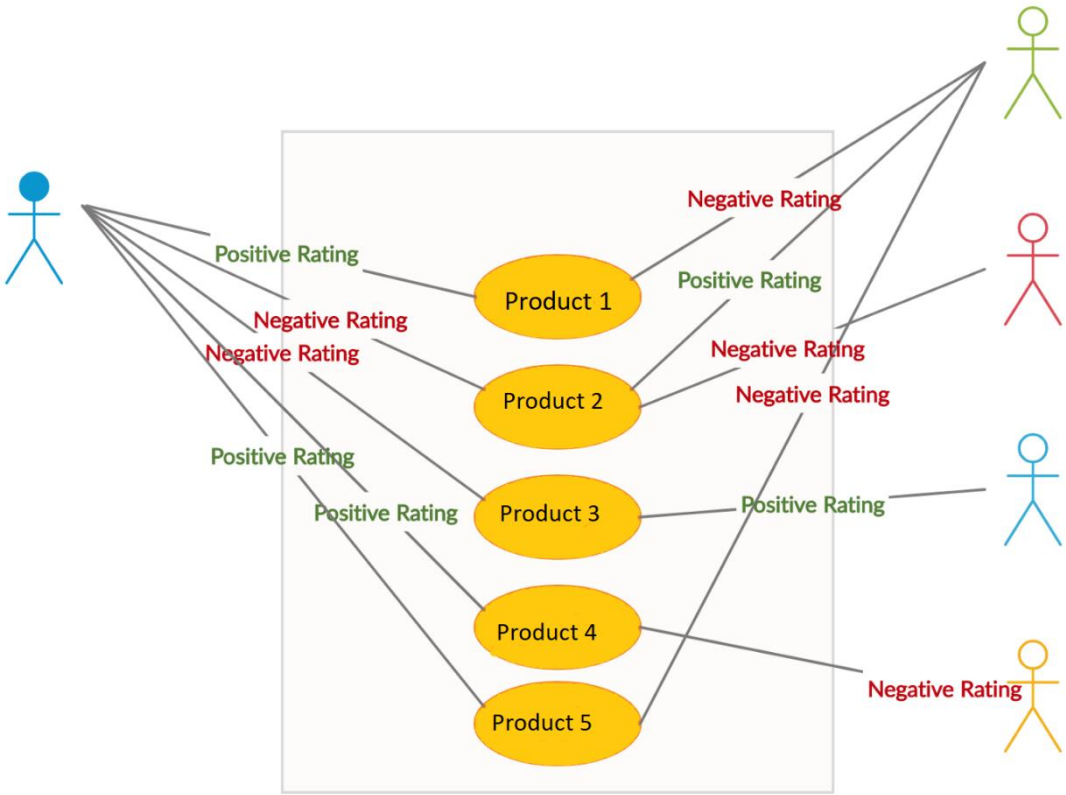
4.2: Class diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



4.3: Use case diagram

Is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.



CHAPTER 5: Implementation Approaches

5.1: Collaborative Approach

One important thing is that in an approach purely based on collaborative filtering, the similarity is not calculated using factors like the age of users, genre of the movie, or any other data about users or items. It is calculated only on the basis of the rating a user gives to an item. Relationships provide recommender systems with tremendous insight, as well as an understanding of customers.

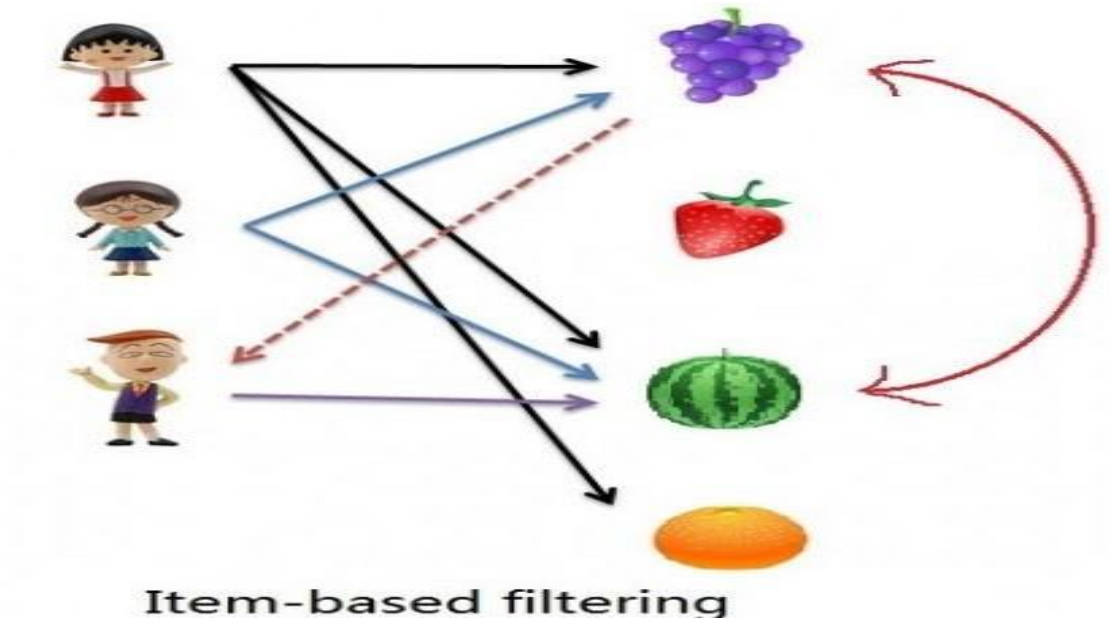
5.2: Types

There are three main types that occur:

1. User-Item Relationship
2. Item-Item Relationship
3. User-User Relationship

5.2.1: Item-Item Relationship

Item-item filtering will take an item, find users who liked that item, and find other items that those users or similar users also liked. It takes items and outputs other items as recommendations. Item-based collaborative filtering was developed by Amazon. In a system where there are more users than items, item-based filtering is faster and more stable than user-based. It is effective because usually, the average rating received by an user to different items. item doesn't change as quickly as the average rating given by an a user to different items. (kaggle, n.d.)



5.3: Algorithm

To build a system that can automatically recommend items to users based on the rating history of the user, the first step is visualizing and preprocessing our data. The second step is to find similarity between products. So, you will require answers of following questions:

1. How to determine the total number of ratings and average rating of each product?
2. How to determine which products are similar to another?

The answer of first question is simple, for getting information about the data you have to analyse the data from where you'll get some basic statistical details for rating like standard deviation, minimum, maximum, etc.. So, using mean() method you'll get the average rating and using count() method you'll get the total number of rating of each product which are important attributes for further calculations.

There are multiple answers available for second question. Collaborative filtering is a family of algorithms where there are multiple ways to find similar products. You can use the correlation between the ratings of a product as the similarity matrix. To find the correlation between the ratings of the product, you need to create a matrix where each column is a product Id and each row contains the rating assigned by a specific user to that product. You can take any product and correlate it with other products, then you'll get the products list which is highly correlated with that product. But correlation alone is not enough. Hence solution to this problem is to retrieve only those correlated products that have at least more than 50 ratings. So, add the total number of rating, which will gives the list of products which are highly correlated and has high rating count.

5.4: Dataset

The dataset that we are going to use for this project is the Electronic product Dataset. We download the dataset from kaggle.com namely “Rating Electronics Data” file, which contains purchase history of electronic products and contains 999 ratings for 17 products purchase by 200 users.

5.4.1: Attribute information

1. User-Id: Every user identified with a unique id
2. Product-Name: Every product identified with a name
3. Brand-Name: Product with specific brand name
4. Rating: Rating of the corresponding product by the corresponding user
5. Review: Review of the product
6. Price: Price of the product 1

CHAPTER 6: RESULTS AND DISCUSSION

6.1: Test Report

Test Report is an important deliverable which is prepared at the end of a Testing project, or rather after Testing is completed. The prime objective of this document is to explain various details and activities about the Testing performed for the Project, to the respective stakeholders like user, Client etc.

Test Report					
Passed	**25**		**Failed**	**2**	
Functions	**Description**	**% TCs Executed**	**% TCs Passed**	**TCs Pending**	**Priority**
datatypes	Check data types of attributes	100%	100%	0	High
dataset	Check all values in our data are not-null	100%	100%	0	High
Visualize data	Check reviews, ratings, rating count of product and arrange	100%	100%	0	High
Preprocessed data	Arrange data in a proper order	100%	100%	0	High
Check best product	Check best product based on highest ratings and rating count	100%	100%	0	High
Find correlation	Check correlation between product	100%	100%	0	High
Search product	Check product	100%	100%	0	High
Recommend product	Check the top related product based on highest correlation , ratings and rating count is recommended	100%	100%	0	High

6.2: User Documentation

let's first import the "projectdataset.xlsx" file and see what it contain

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
```

```
In [2]: Elect_data = pd.read_excel(r"C:\ProjectData.xlsx")
        Elect_data.head()
```

Out[2]:

	User_Id	Product_Name	Brand_Name	Price	Rating	Reviews
0	683435088	"CLEAR CLEAN ESN" Sprint EPIC 4G Galaxy SPH-D7...	Samsung	199.99	5	I feel so LUCKY to have found this used (phone...
1	683435089	"CLEAR CLEAN ESN" Sprint EPIC 4G Galaxy SPH-D7...	Samsung	199.99	4	nice phone, nice up grade from my pantach revu...
2	683435090	"CLEAR CLEAN ESN" Sprint EPIC 4G Galaxy SPH-D7...	Samsung	199.99	5	Very pleased
3	683435091	"CLEAR CLEAN ESN" Sprint EPIC 4G Galaxy SPH-D7...	Samsung	199.99	4	It works good but it goes slow sometimes but i...
4	683435092	"CLEAR CLEAN ESN" Sprint EPIC 4G Galaxy SPH-D7...	Samsung	199.99	4	Great phone to replace my lost phone. The only...

In the script above we use the read-excel() method of the Pandas library to read the "projectdataset.xlsx" file. Next, we call the head() method from the data frame object returned by the read-excel() function, which will display the first five rows of the dataset. You can see from the output that the "projectdataset.xlsx" file contains the User Id, ProductName, Brand Name, Price, Ratings, and Reviews attributes. Each row in the dataset corresponds to one rating. The user-Id column contains the ID of the user who left the rating. The ProductName and Brand Name column contains the name of the product and brand. The Price column contains price of the product, the rating column contains the rating left by the user. Ratings can have values between 1 and 5. And finally, the Reviews refers to the comments which the user left.

```
In [3]: Elect_data.shape
Out[3]: (999, 6)

In [4]: #Check the datatypes
        Elect_data.dtypes
Out[4]: User_Id           int64
        Product_Name     object
        Brand_Name       object
        Price           float64
        Rating            int64
        Reviews          object
        dtype: object

In [5]: Elect_data.info()
        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 999 entries, 0 to 998
        Data columns (total 6 columns):
         #   Column        Non-Null Count  Dtype
        ---  ------        --------------  -----
         0   User_Id       999 non-null    int64
         1   Product_Name  999 non-null    object
         2   Brand_Name    999 non-null    object
         3   Price         999 non-null    float64
         4   Rating        999 non-null    int64
         5   Reviews       999 non-null    object
        dtypes: float64(1), int64(2), object(3)
        memory usage: 47.0+ KB
```

We called info() method for checking any null entry in the dataset. But, here all entries are non-null. So, dataset set is full fil. Now, using describe() method to view some basic statistical details for rating like count, mean, standard deviation, etc.

```
In [6]: Elect_data.describe()['Rating']
Out[6]: count    999.000000
        mean       3.724725
        std        1.562107
        min        1.000000
        25%        2.000000
        50%        5.000000
        75%        5.000000
        max        5.000000
        Name: Rating, dtype: float64
```

```
In [7]: #Find the minimum and maximum ratings
        print('Minimum rating is: %d' %(Elect_data.Rating.min()))
        print('Maximum rating is: %d' %(Elect_data.Rating.max()))

        Minimum rating is: 1
        Maximum rating is: 5
```

Now let's take a look at the average rating of each product. To do so, we can group the dataset by the brand of the product and then calculate the mean of the rating for each brand. We will then display the first five brand name along with their average rating using the head() method.

```
In [8]:
Elec_data.groupby('Brand_Name')['Rating'].mean().head()

Out[8]:

Brand_Name
Apple                       3.650602
BlackBerry                  3.989691
Cedar Tree Technologies     2.000000
HTM                         4.029412
Huawei                      4.320755
Name: Rating, dtype: float64
```

You can see that the average ratings are not sorted. Let's sort the ratings in the descending order of their average ratings:

```
In [9]:
rating_mean= Elec_data.groupby('Brand_Name')['Rating'].mean().sort_values(ascending=False)
rating_mean.head()

Out[9]:

Brand_Name
Phone Baby    5.000000
Motorola      4.678571
Huawei        4.320755
OtterBox      4.233333
Nokia         4.205479
Name: Rating, dtype: float64
```

The product brand have now been sorted according to the ascending order of their ratings. However, there is a problem. A product brand can make it to the top of the above list even if only a single user has given it five stars. Therefore, the above stats can be misleading. Normally, a product brand which is really a good one gets a higher rating by a large number of users. Let's now plot the total number of ratings for a product brand:

In [10]:

```
rating_count= Elec_data.groupby('Brand_Name')['Rating'].count().sort_values(ascending=False
rating_count.head()
```

Out[10]:

```
Brand_Name
Lenovo        163
Jethro        117
BlackBerry     97
LG             93
Ulefone        83
Name: Rating, dtype: int64
```

In [11]:

```
print("Total data ")
print("\nTotal no of ratings :",Elec_data.shape[0])
print("Total No of Users   :", len(pd.unique(Elec_data.User_Id)))
print("Total No of products  :", len(pd.unique(Elec_data.Brand_Name)))
```

```
Total data

Total no of ratings : 999
Total No of Users   : 200
Total No of products  : 17
```

We know that both the average rating per brand and the number of ratings per brand are important attributes. Create a new data frame that contains both of these attributes.

```
In [12]: ratings_mean_count = pd.DataFrame(Elect_data.groupby('Brand_Name')['Rating'].mean
         ratings_mean_count['rating_counts'] = pd.DataFrame(Elect_data.groupby('Brand_Name
         ratings_mean_count.head()
```

Out[12]:

Brand_Name	Rating	rating_counts
Apple	3.650602	83
BlackBerry	3.989691	97
Cedar Tree Technologies	2.000000	4
HTM	4.029412	34
Huawei	4.320755	53

```
In [13]: ratings_counts=ratings_mean_count.iloc[:,1].values
         ratings_counts
```

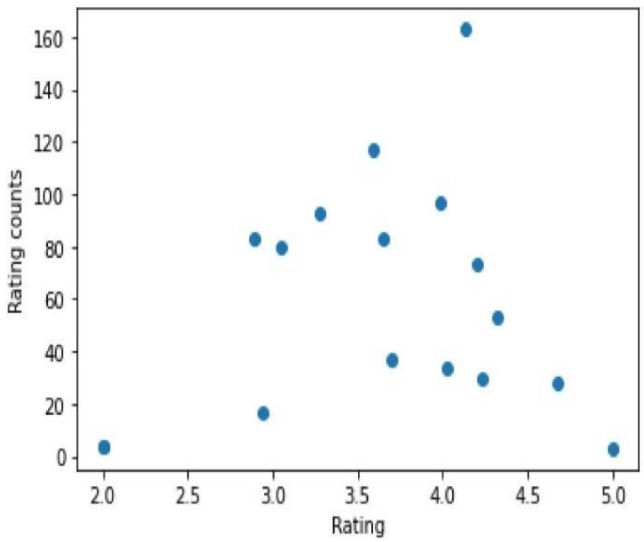
```
Out[13]: array([ 83,  97,   4,  34,  53,   4, 117,  93, 163,  28,  73,  30,   3,
                 37,  80,  83,  17], dtype=int64)
```

```
In [14]: Ratings=ratings_mean_count.iloc[:,0].values
         Ratings
```

```
Out[14]: array([3.65060241, 3.98969072, 2.        , 4.02941176, 4.32075472,
                2.        , 3.58974359, 3.27956989, 4.12883436, 4.67857143,
                4.20547945, 4.23333333, 5.        , 3.7027027 , 3.05      ,
                2.89156627, 2.94117647])
```

We know that brand with a higher number of ratings usually have a high average rating as well since a good product is normally well-known and a well known product is purchase by a large number of people, and thus usually has a higher rating.

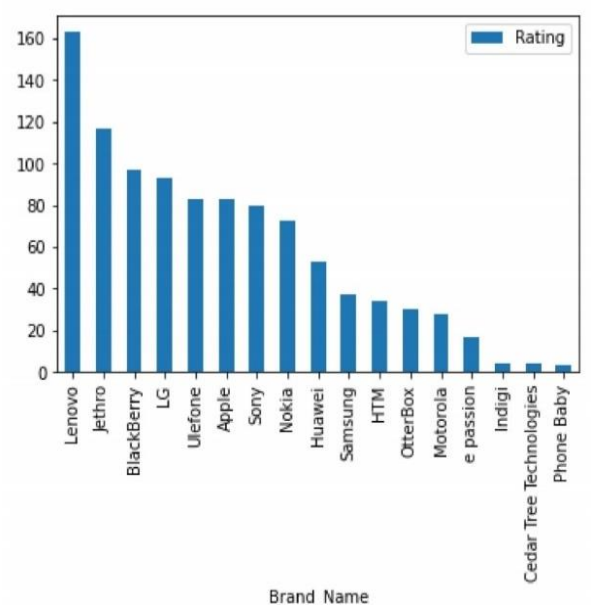
```
In [15]: plt.xlabel('Rating')
         plt.ylabel('Rating counts')
         plt.scatter(Ratings,ratings_counts)


         plt.show()
```



Let's plot a histogram for the number of ratings in descending order represented by the rating counts.

```
In [16]: popular_products = pd.DataFrame(Elect_data.groupby('Brand_Name')['Rating'].count(
         most_popular = popular_products.sort_values('Rating', ascending=False)
         most_popular.head(30).plot(kind = "bar")
```

```
Out[16]: <AxesSubplot:xlabel='Brand_Name'>
```



Here, I will use the correlation between the ratings of a product as the similarity metric. To find the correlation between the ratings of the product, I need to create a matrix where each column is a brand name and each row contains the rating assigned by a specific user to that product. Bear in mind that this matrix will have a lot of null values since every movie is not rated by every user.

```
In [17]: user_Brand_rating =Elect_data.pivot_table(index='User_Id', columns='Brand_Name',
         user_Brand_rating.head()
```

```
Out[17]:
```

Brand_Name	Apple	BlackBerry	Cedar Tree Technologies	HTM	Huawei	Indigi	Jethro	LG	Lenovo	Motorol
User_Id										
683435088	0.0	0	0	0	0	0	3.0	5.0	0.0	
683435089	0.0	0	0	0	0	0	5.0	3.0	0.0	
683435090	0.0	0	0	0	0	0	5.0	3.0	0.0	
683435091	0.0	0	0	0	0	0	5.0	5.0	0.0	
683435092	0.0	0	0	0	0	0	5.0	3.0	0.0	

Assuming the customer search for product “Jethro”

```
In [18]: Jethro_ratings = user_Brand_rating['Jethro']
         Jethro_ratings.head()

Out[18]: User_Id
         683435088    3.0
         683435089    5.0
         683435090    5.0
         683435091    5.0
         683435092    5.0
         Name: Jethro, dtype: float64
```

Now let’s retrieve all the brand name that are similar to “Jethro”. We can find the correlation between the user ratings for the “Jethro” and all the other brands using corrwith() function

```
In [19]:

brand_like_jethro = user_brand_rating.corrwith(Jethro_ratings,axis=0,drop=False,method='pea
brand_like_jethro.head()

Out[19]:

Brand_Name
Apple                    -0.429777
BlackBerry               -0.014784
Cedar Tree Technologies   0.084185
HTM                       0.361630
Huawei                    0.399466
dtype: float64
```

In the above script, we first retrieved the list of all the brand name related to “Jethro” along with their correlation value, using corrwith() function.

Next, we created a data frame that contains brand name and correlation columns.

```
In [20]: corr_Jethro = pd.DataFrame(Brand_like_Jethro , columns=['Correlation'])
         corr_Jethro.dropna(inplace=True)
         corr_Jethro.head()
```

Out[20]:

Brand_Name	Correlation
Apple	-0.429777
BlackBerry	-0.014784
Cedar Tree Technologies	0.084185
HTM	0.361630
Huawei	0.399466

Let's sort the product brand in descending order of correlation to see highly correlated products at the top.

```
In [21]:
corr_jethro.sort_values('Correlation', ascending=False).head()
```

Out[21]:

Brand_Name	Correlation
Jethro	1.000000
Nokia	0.611878
Huawei	0.399466
OtterBox	0.369417
Samsung	0.364072

From the output you can see that the brands that have high correlation with "Jethro" are not very well known. This shows that correlation alone is not a good metric for similarity because there can be a user who purchase "Jethro" and only one other product and rated both of them as 5. A solution to this problem is to retrieve only those correlated product that have at least more than 50 ratings. To do so, will add the rating

counts column from the rating mean count data frame to our correlation with Jethro data frame.

```
In [22]: corr_Jethro = corr_Jethro.join(ratings_mean_count['rating_counts'])
         corr_Jethro.head()
```

Out[22]:

Brand_Name	Correlation	rating_counts
Apple	-0.429777	83
BlackBerry	-0.014784	97
Cedar Tree Technologies	0.084185	4
HTM	0.361630	34
Huawei	0.399466	53

You can see that the product “Cedar Tree Technologies”, which has the highest correlation has only four ratings. This means that only four users gave same ratings to “Jethro”, “Cedar Tree Technologies”. However, we can deduce that a product cannot be declared similar to the another product based on just four ratings. This is why we added “rating counts” column. Let’s now filter product brand correlated to “Jethro”, that have more than 50 ratings.

```
In [23]: corr_Jethro[corr_Jethro['rating_counts']>50].sort_values('Correlation', ascending
```

Out[23]:

Brand_Name	Correlation	rating_counts
Jethro	1.000000	117
Nokia	0.611878	73
Huawei	0.399466	53
Sony	0.347884	80
BlackBerry	-0.014784	97

Here are the top products to be displayed by the recommendation system which are “Nokia”, “Sony”, “Huawei” to the above customer based on high correlation and rating count.

Similarly, if you search for product “Lenovo”

```
In [24]: Lenovo_ratings = user_Brand_rating['Lenovo']
         Lenovo_ratings
```

```
Out[24]: User_Id
         683435088    0.0
         683435089    0.0
         683435090    0.0
         683435091    0.0
         683435092    0.0
                     ...
         683435283    5.0
         683435284    5.0
         683435285    5.0
         683435286    2.0
         683435287    1.0
         Name: Lenovo, Length: 200, dtype: float64
```

```
In [25]: Brand_like_Lenovo = user_Brand_rating.corrwith(Lenovo_ratings,axis=0,drop=False,
         Brand_like_Lenovo.head()
```

```
Out[25]: Brand_Name
         Apple                       0.435434
         BlackBerry                  0.086862
         Cedar Tree Technologies    -0.118554
         HTM                        -0.395286
         Huawei                     -0.421819
         dtype: float64
```

```
In [26]: corr_Lenovo = pd.DataFrame(Brand_like_Lenovo , columns=['Correlation'])
         corr_Lenovo.dropna(inplace=True)
         corr_Lenovo.head()
```

Out[26]:

Brand_Name	Correlation
Apple	0.435434
BlackBerry	0.086862
Cedar Tree Technologies	-0.118554
HTM	-0.395286
Huawei	-0.421819

In [27]:
```
corr_Lenovo.sort_values('Correlation', ascending=False).head()
```

Out[27]:

Brand_Name	Correlation
Lenovo	1.000000
Ulefone	0.601293
Apple	0.435434
Motorola	0.387328
LG	0.270048

In [28]:
```
corr_Lenovo = corr_Lenovo.join(ratings_mean_count['rating_counts'])
corr_Lenovo.head()
```

Out[28]:

Brand_Name	Correlation	rating_counts
Apple	0.435434	83
BlackBerry	0.086862	97
Cedar Tree Technologies	-0.118554	4
HTM	-0.395286	34
Huawei	-0.421819	53

In [29]:
```
corr_Lenovo[corr_Lenovo['rating_counts']>50].sort_values('Correlation', ascending
```

Out[29]:

Brand_Name	Correlation	rating_counts
Lenovo	1.000000	163
Ulefone	0.601293	83
Apple	0.435434	83
LG	0.270048	93
BlackBerry	0.086862	97

Here, you can see that recommendation system recommended “Ulefone”, “Apple”, “LG” products. In this way, if you search for any other product, then recommendation system will recommend you products based on correlation and rating counts.

CHAPTER 7: Conclusions

7.1: Conclusion:

- Project is a courteous initiative to meet the caters needs. The objective of the software planning is to provide a framework that enables the people to book caters.
- We made a technology survey to fulfil the current trends of technology. The project is structured according to today's technology need. As per user requirements and demand, we are trying to do our best. Finally, it is concluded that we have made effort on following points:
- A description of the background and context of the project and its relation to work already done in the area.
- Made statement of the aims and objectives of the project.
- The description of purpose and applicability
- We define the problem on which we are working in the project
- We describe the requirements specifications of the system and the actions that can be done on these things.
- We understand the problem domain and produce the model of the system.
- We included features and operations in details.
- We discussed the three traditional recommendation techniques and highlighted collaborative recommendation technique.
- Various learning algorithms used in generating recommendation models and correlation metrics used in measuring the performance of recommendation algorithms were discussed.
- Finally, we are recommending the products to the customer based on the rating according to test cases.

7.1.1: Significance of the system

- 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 a particular user.
- We don't need domain knowledge because the embeddings are automatically learned. The model can help users discover new interests. In isolation, the ML system may not know the user is interested in a given item, but the model might still recommend it because similar users are interested in that item.
- Collaborative Filtering provides strong predictive power for recommender systems, and requires the least information at the same time.

7.2: Limitation of the system

- Collaborative filtering can lead to some problems like cold start for new items that are added to the list. Until someone rates them, they don’t get recommended.
- Data sparsity can affect the quality of user-based recommenders and also add to the cold start problem mentioned above.
- Scaling can be a challenge for growing datasets as the complexity can become too large. Item-based recommenders are faster than user-based when the dataset is large.
- With a straightforward implementation, you might observe that the recommendations tend to be already popular, and the items from the long tail section might get ignored.

7.3: Future scope of the project

- Neural Networks and Deep Learning have been all the rage the last couple of years in many different fields, and it appears that they are also helpful for solving recommendation system problems.
- One of the benefits of Deep Learning is similar to matrix factorization, in that there is an ability to derive latent attributes.
- Deep Learning, however, can make up for some of the weaknesses of matrix factorization such as the inability to include time in the model — which standard matrix factorization isn't designed for. Deep Learning, however, can utilize Recurrent Neural Networks which are specifically designed for time and sequence data.
- Recommender systems can be a very powerful tool in a company's arsenal, and future developments are going to increase business value even further. Some of the applications include being able to anticipate seasonal purchases based on recommendations, determine important purchases, and give better recommendations to customers which can increase retention and brand loyalty.
- Most businesses will have some use for recommender systems.

Bibliography

journaldev. (n.d.). Retrieved from journaldev: https://www.journaldev.com/29055/python-pandas-module-tutorial

kaggle. (n.d.). Retrieved from kaggle: https://www.kaggle.com/code/saurav9786/recommender-system-using-amazon-reviews/notebook

kaggle. (n.d.). Retrieved from kaggle: https://www.kaggle.com/code/saurav9786/recommender-system-using-amazon-reviews/notebook

matplotlib. (n.d.). Retrieved from matplotlib: https://matplotlib.org/

mygreatlearning. (n.d.). Retrieved from mygreatlearning: https://www.mygreatlearning.com/blog/python-numpy-tutorial/#:~:text=NumPy%2C%20which%20stands%20for%20Numerical,stands%20for%20'Numerical%20Python'.

towardsdatascience. (n.d.). Retrieved from towardsdatascience: https://towardsdatascience.com/recommender-systems-item-customer-collaborative-filtering-ff0c8f41ae8a?gi=8fa52e754361

wikipedia. (n.d.). Retrieved from wikipedia: https://en.wikipedia.org/wiki/Recommender_system