Recommendation System for the Purchase Data of the Shopping Arcades

by Naqash Haider Yasir Irfan

Submission date: 25-Jul-2020 08:22PM (UTC+0500)
Submission ID: 1361970453
File name: for_plagiarism_report1-converted.pdf (1.58M)
Word count: 7418
Character count: 37909

INTRODUCTION

1. INTRODUCTION

Recommendation system aims to predict customer's interest and recommend products and items that are quite likely are interesting and priority for them. Also, a Recommendation System refer to a system that can anticipate future inclination of a lot of things and items for a user suggest the top items that will ultimately be beneficial for the owner of the shopping arcades. This kind of recommendation systems are used in a variety of different sectors especially in supermarkets like: -

- ALDI
- Amazon
- E-Bay

In the contemporary world, super stores and increasing in the market. So basically, we aimed to develop this recommendation system to target small shopping arcades to use this system as a marketing tool to increase their sale for competing with high level super stores.

When a customer buys some specific products or items from any shopping arcade the recommendation system will filter the customer's interest, choice, product types, previous products and purchases. Based on following calculations the recommendation system will help the owner to categorize and arrange the products in a way that customers can easily access them.

1.2 BRIEF

Here is the brief introduction of Recommendation System.

1.2.1 APRIORI ALGORITHM [1]

The April algorithm is used to find a set of repetitive items purchased together with transactional data. The Support, Confidence and lift can be used to create rules for shared and freely rules for the frequently purchased items that are bought together. For Recommendation in Market basket and E-Commerce mostly Apriori Algorithm is used. Customers who buy the pattern decide to push the Recommendations using the items in the cart as a parameter.

The project will help the Buyer's to easily find the products. Machine Learning Algorithms will help in the buyers in following.

- Recommend items to Owner to change rack order.
- Increase in income.

- Special day Recommendation.
- Market need.
- Seasonal base Recommendation.

1.3 RELEVANCE TO COURSE MODULE

Our project is related with different subjects like Machine Learning, Artificial Intelligence in machine we have learned Apriori Algorithm and in Artificial Intelligence we have learned Python.

1.4 PROJECT BACKGROUND

Our Recommendation system is specially designed for non-tech managers of the shopping arcades. Collaborative Filtering of this system will help the manager to calibrate its product category according to the preferences of its customers.

Any Shopping arcade sells thousands of items daily and with the help of this recommendation system, the owners can easily categorize the data about every product and change the arrangement of items according to user interest.

1.4.1 ADVANTAGES

Our recommendation system has the following benefits:

- Recommend items to admin to change rack order
- Easiness
- Increase in income
- Special day Recommendation
- Market need
- Seasonal Based Recommendation

1.4.2 TOOLS AND TECHNOLOGY

For the Development of our system we will use following Tools and Technology:

'**Anaconda**' is the best open source dispersion for the python to perform Data Science and Machine Learning.

1.5 LITERATURE REVIEW

As we have referenced before, we are developing our system by analyzing the absolute best Recommendation System like Amazon, Alibaba, YouTube, Netflix and so on. Our Project primarily centers around the Recommendation System algorithms used in these Systems.

1.5.1 AMAZON [2]

The achievements of the Amazon is all because of its inventive thoughts and useage of those thoughts into its Algorithms. Amazon is utilizing blend of the Algorithms for example Client Based Collaborative Filtering and Item Based Collaborative Filtering further more with the blend of these calculations there is a type of AI is included which causes the framework to improve its Recommending Phenomena every once in a while. Notable Algorithm associated with Amazon are following:

- Affinity Analysis Algorithm for Market-Based Analysis
- Mixture of AI libraries for the system Intelligence

The excellence of Amazon Algorithms is that it forms up to multiple times greater size of requests and it requires some investment during the procedure occurs when contrasted with other Recommendation Systems available.

1.5.2 ALIBABA [3]

Alibaba Utilize their restrictive proposal calculations so as to more readily serve the clients with the items they will undoubtedly like. There are significantly more advantages as well, which we cover in the next blogs.

They have set up and designed these algorithms so that these algorithms are fundamentally helping the Alibaba to support incomes, CTRs, transformations, and other significant measurements. In addition, they can positively effect on the client experience too, which converts into measurements that are more diligently to gauge yet are regardless of much significance to online organizations, for example, consumer loyalty and maintenance.

Alibaba is utilizing 'Hybrid Recommendation System based on personalized Algorithm'. This set of Algorithm is causing Alibaba to execute Artificial Intelligence to their selling System.

1.5.3 FLIPCART [4]

Here's a run of the mill venture when you need to purchase any item from Flipkart suppose you looked for 'Pants', for instance, click on an item that gets your interest, and from that point on, You will depend on 'Comparative items' Recommendations to lead you toward your preferred result. What's more, you are not the only one in this, Recommendation Systems are utilizing interior information just as client examines show this is one of the most well-known excursion ways of our clients, particularly when purchasing from exploratory classes, for example, Fashion and Home where there is enormous choice and individuals for the most part don't have explicit item or brand on mind.

Flipkart utilizes 'Hybrid of Content and collaborative Filtering' strategies to create Similar Product Recommendations'. The content coordinating is done over item characteristics and pictures in the list, and the Collaborative Filtering is applied over clients' peruse information (like item site visits, list of things to get, add to cart and so on.) to locate the most regularly co-perused items for a given item. The positioned rundown of comparable items dependent on importance is gotten from joining these numerous sources.

1.5.4 NEW YORK TIMES [5]

The New York Times distributes in excess of 300 articles, blog entries and intelligent stories daily for its clients or clients.

In the Recommendation segment of the New York Times for example "**Recommended for You**" there are two sorts of sifting or proposal calculations that are giving the practically precise outcomes to their clients.

1.5.4.1 CONTENT BASED FILTERING [6]

This Filtering System utilizes keywords and labels to make Recommendations. It labels the articles and it additionally gets to the perusing history of recent days; in view of that history the Recommendations are performed. Since this method depends on a Content model, it is a piece of a more extensive class of Content based Recommendation Algorithm.

1.5.4.2 COLLABORATIVE FILTERING [7]

These filters surface articles dependent on what related pursuers have perused, for this situation the Similarity is accessed by getting to their authentic information.

1.6 ANALYSIS FROM LITERATURE REVIEW

Application Name	Weakness	Proposed Project Solution
Online Recommendation Systems (Amazon.com, Netflix, Shopify, Alibaba etc.)	These systems are online.	We are developing the system which is going to work on desktop without internet connection.

Table 1: Analysis from Literature Review

1.7 METHODOLOGY AND SOFTWARE LIFECYCLE

The development of this project will follow Agile Model.[8]

1.7.1 AGILE MODEL

Figure 1: Methodology and Software Life Cycle

Following are the phases in the agile model are as follows:

- Requirements gathering.
- Design the requirements.
- Construction/ iteration.
- Testing/ Quality assurance.
- Deployment.
- Feedback.

1.7.2 RATIONALE BEHIND SELECTED MODEL

- Stakeholder Engagement.
- Transparency.
- Early and Predictable Delivery.
- Predictable Costs and Schedule.
- Allows for Change.
- Focuses on Business Value.
- Focuses on Users.
- Improves Quality.

19

PROBLEM DEFINITION

2. PROBLEM DEFINITION

2.1 PROBLEM STATEMENT

As the number of shopping arcades are increasing in number and customers there have their own perception and choice of certain products and items, which is an important aspect for the growth of these shopping arcades. It is important to address the issues that will attract the customers and that help in increasing sale of the business of these shopping arcades.

2.2 DELIVERABLE AND DEVELOPMENT REQUIREMENTS

Despot application will be built to help non-tech managers of shopping cart. Product will be delivered in three phases, as version 1-3.

Version 1

1.1. Item Recommendation.

1.1.1 Daily basis

1.1.2 Weekly

1.1.3 Monthly

Version 2

2.1. Seasonal Based Recommendation

2.2. Market Need

Version 3

3.1. Special Event.

3.2. Minimum items sale.

Requirement Analysis

3. REQUIREMENT ANALYSIS

3.1 USE CASE DIAGRAM

Use case diagrams are used to describe the set of action that system should or can perform in collaboration with one or more external of the system.

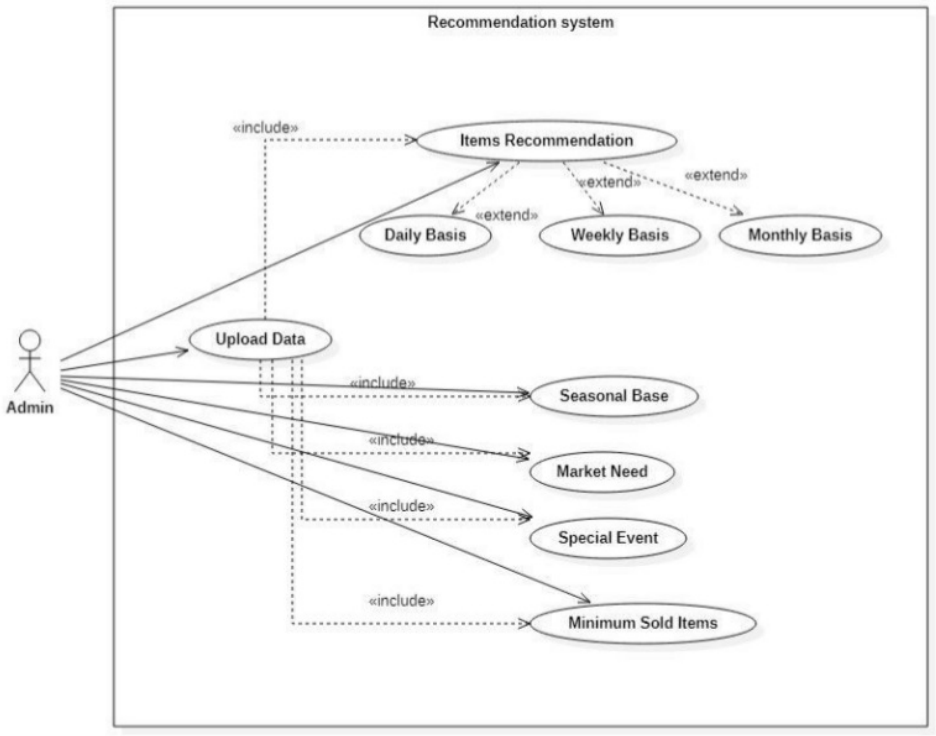

Figure 2: Use Case Diagram

3.2 DETAILED USE CASE DIAGRAM

3.2.1 UPLOAD DATA

Use case Name: Upload Data

Actor(s): Admin.

Type: Primary and essential.

Description: The admin can upload data and if data is in correct format after processing then it can be accepted otherwise not.

Use cases: Admin must upload data for further process /requirement.

3.2.2 ITEM RECOMMENDATION

Use case Name: Item Recommendation.

Actor(s): Admin.

Type: Primary and essential.

Description: After uploading data admin can check the recommendation according to data that can be uploaded (It can be on daily, weekly and monthly basis also).

Use cases: Admin can check the recommendation and change the rack order according to user interest.

3.2.3 SEASONAL BASED RECOMMENDATION

Use case Name: Seasonal Based Item Recommendations

Actor(s): Admin.

Type: Primary and essential.

Description: After uploading data admin can check the recommendation and if the data is more than 1 year then it can check the Seasonal base recommendation.

Use cases: Admin can check the recommendation and after that it can maintain, and update is stock according to user interest in previous season.

3.2.4 MARKET NEED

Use case Name: Market need.

Actor(s): Admin.

Type: Primary and essential.

Description: After uploading data admin can check the recommendation that which items are sold most and have more market need.

Use cases: Admin can check the recommendation and after that it can maintain, and update is stock according to user interest and market need.

3.2.5 SPECIAL EVENTS

Use case Name: Special Event.

Actor(s): Admin.

Type: Primary and essential.

Description: After uploading data admin can check the recommendation that which items are sold in previous year on that special day or event (Eid, Christmas, Independence Day etc.)

Use cases: Admin can check the recommendation and after that it can maintain, and update is stock according to that special day or event.

3.2.6 MINIMUM SOLD ITEMS

Use case Name: Minimum sold items

Actor(s): Admin

Type: Primary and essential

Description: After uploading data admin can check the recommendation that which items are sold in minimum level and how to sell these items from mart or arcade (like make discount on that items)

Use cases: Admin can check the recommendation and after that it can maintain stock and take some decision how to sell that items.

3.3 FUNCTIONAL REQUIREMENTS

Name	FR: 1 Upload Data
Summary	Through Upload Data Admin can upload the data of their sold items.
Rationale	With the recommendation the admin comes to know about frequently sold item and most related sold product.
Requirements	The system will recommend frequently bought items and most related product.

Table 2: FR:1 Upload Data

Name	FR: 2 Validate Data
Summary	The data which is uploaded should be validate otherwise data cannot be uploaded for further process.
Rationale	If the data is according to our required format then accept, otherwise system will show error.

Requirements	The main requirement is that the admin can upload data according to given format.

Table 3: FR: 2 Validate Data

Name	FR: 3 Converting data into required format.
Summary	The data which is uploaded is converting into required format and then ready for further process.
Rationale	If the data is according to our required format then accept, otherwise system will show error.
Requirements	The main requirement is that the admin can upload data according to given format.

Table 4: FR: 3 Converting data into required format.

Name	FR: 4 Item Recommendations to admin to change Rack order.
Summary	Admin will receive the Recommendations for the items that are sold most frequently and the items which customer may like to buy.
Rationale	According to these Recommendations the admin can change the order of rack of different items.
Requirements	The Recommendation will be sent to Admin based on user interest and the items that are sold most regularly.

Table 5: FR: 3 Item Recommendations

Name	FR: 4 Daily basis item Recommendations to admin.
Summary	Admin will receive the Recommendation for Daily Sold Items, items which are sold most and the item which are sold minimum. .
Rationale	With the help of the se Recommendations the admin can change the order of rack of different items.
Requirements	The Recommendation will be sent to Admin based on user interest and Daily Demanding items.

Table 6: FR: 4 Daily basis item Recommendations

Name	FR: 5 Monthly Basis Item Recommendations to admin.
Summary	Admin will receive the Recommendation for Monthly Sold Items, items which are sold most and the item which are sold minimum. .
Rationale	With the help of these Recommendations the admin can change the order of rack of different items.
Requirements	The Recommendation will be sent to Admin based on user interest and Monthly Demanding items.

Table 7: FR: 5 Monthly Basis Item Recommendations

Name	FR: 6 Market Need.
Summary	Admin will receive the list of items that are sold most and the list of items that are sold minimum and maximum sold item as well.
Rationale	With recommendation the admin can maintain his stock and improve his/her sale.
Requirements	All Recommendations according to Market Need will be sent to Admin based on items that are sold most and the list of items that are sold minimum.

Table 8: FR: 6 Market Need

Name	FR: 7 Seasonal Recommendation
Summary	Admin will receive Recommendations for the Items that are sold mostly in previous Season
Rationale	With recommendation the admin can replace the item of coming season that the customer purchased in previous years or previous seasons (Summer / Winter).
Requirements	Recommendations will be sent to Admin based on user interest, and the items that are mostly sold in previous season.

Table 9: FR: 7 Seasonal Item Recommendations

Name	FR: 8 Special Day or Event.
Summary	Admin will receive the recommendations according to that specific Special Day or Event and notification for items.

Rationale	With recommendation the admin can change the order of rack of different items.
Requirements	Recommendations will be sent to Admin based on the items which are sold mostly on Specific Special Day or Event.

Table 10: FR: 8 Special Day or Event

3.4 NON-FUNCTIONAL REQUIREMENTS

3.4.1 USABILITY

This Recommendation System is very simple and easy to use because we make simple and friendly interface especially for non-tech Admin because most of the admin are not so much educated so we minimize the complexity for user by making simple and attractive user interface.

3.4.2 RELIABILITY

Our system runs very smoothly no crash in case any query. When user use, it they find it what they need. Our data regarding recommendation is accurate because it pass/flow from different process.

3.4.3 DURABILITY

The life span of our app data is for 5 year. Every 5 year our app data is totally change and replaced by new data due environmental and climate issue.

3.4.4 APPEARANCE

Our Recommendation system interface is quite simple, attractive and very suitable for admin. The main theme of interface is simplicity not too much color involved.

3.4.5 AVAILABILITY

Our system is available 24/7. Users use it any time.

3.4.6 SAFETY

Our system interface is not causing any kind of eyesight, headache or any other health issues. The data we display to user is safe not harmful.

3.4.7 SECURITY

Any kind of data loss is not occurred. The data we display to the user is secure and taken from user storage.

3.4.8 SUSTAINABILITY

Every module of the system is stable mean user does not suffer from that one module run perfectly and another module not run smoothly. Every feature in the system performing in the same way the data display is stable.

3.4.9 USEFULLNESS

This system design for owner or admin those suffer from many marketing, managing and advertising issues like which items sell more with items. So, our system provided very informative and gain able information to the user.

3.4.10 DELIGHTFULLNESS

Recommendation system is not only an interface, but it is an revolution that delight the arcade owner/admin.it is not just an system it is platform that gives a power to the owner to enhance their business.

3.4.11 PERFORMANCE

Being a user If I perform any task, then the requires response should be very fast, so by keeping in view this demand of user we cope this issue and we try to keep minimum query time.

3.4.12 EFFICIENCY

This system uses minimum resources of your device.

3.4.13 MAINTAINABILITY

Any sort of change in functionalities, design in system for Admin is easy because latest code re factoring and other maintainable technique we use like every module has own package so easily change any sort of error, improvement etc.

3.4.14 UNDERSTANDABLE

Easy understand the design, code and overall architecture of software.

DESIGN
AND
ARCHITECTURE

4. DESIGN AND ARCHITECTURE

4.1 ACTIVITY DIAGRAM

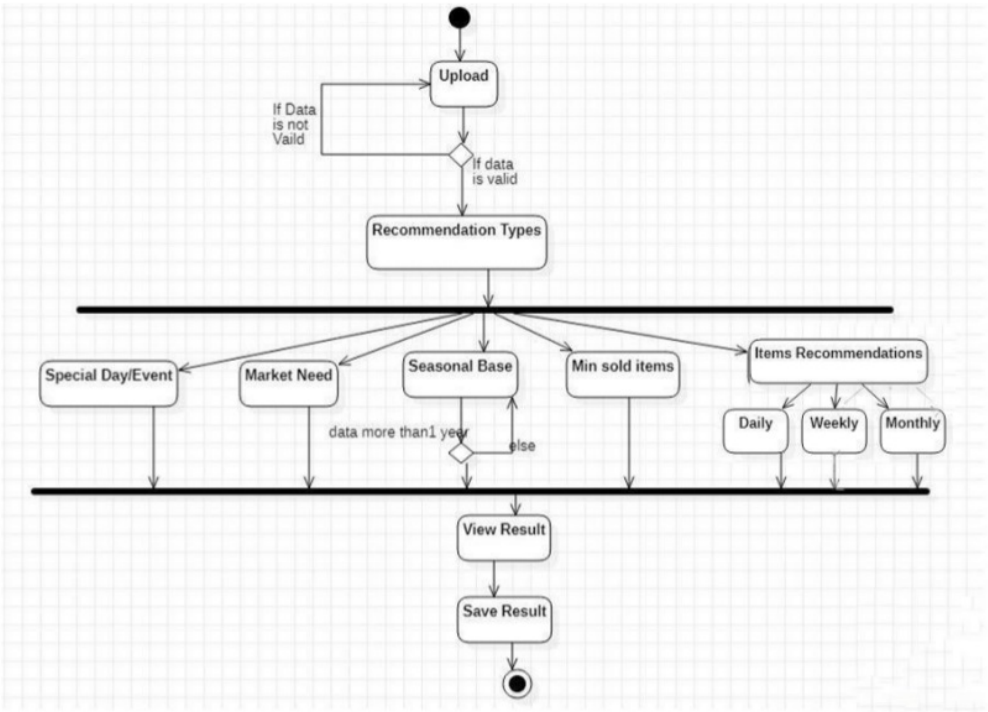

Figure 3: Activity Diagram

In Figure 1: Activity Diagram we describe the flow of our system firstly admin can upload the data if data is according to our means then system will accept for further process otherwise system ask for correct data.

After uploading correct data system will process the data with different algorithms and give the different type of recommendation. Seasonal based recommendation will work when the data is more than 1-year transaction. If the data is available for more than 1 year of transaction system will recommend items according to these transaction and user have option of just view result or save it also.

4.2 DATA FLOW DIAGRAM

4.2.1 LEVEL 0

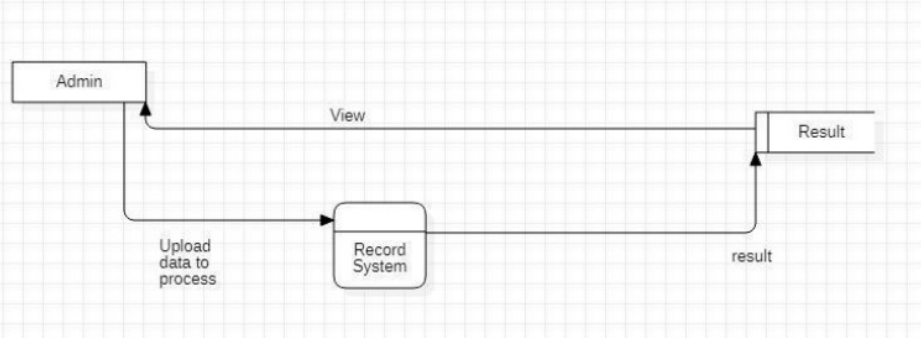

Figure 4: Level 0

4.2.2 LEVEL 1

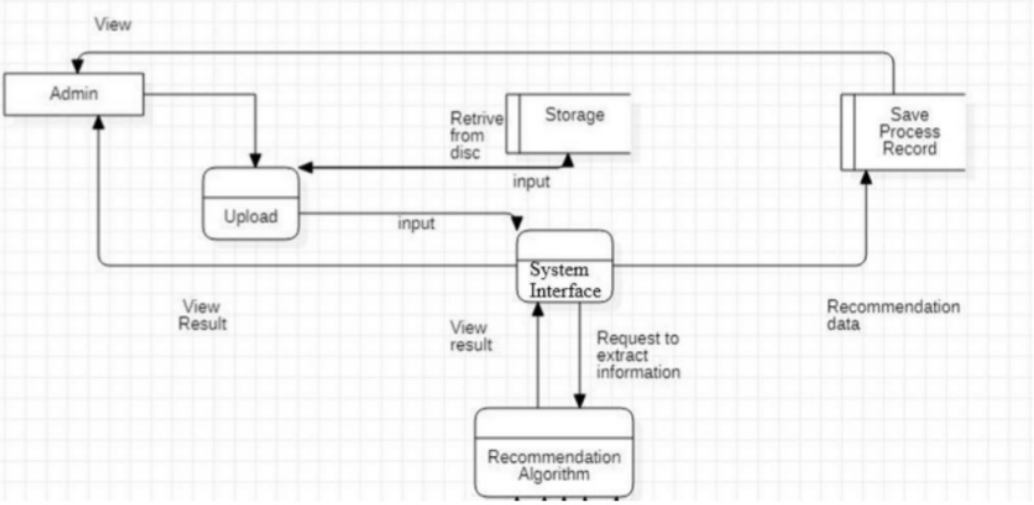

Figure 5: Level 1

4.2.3 LEVEL 2

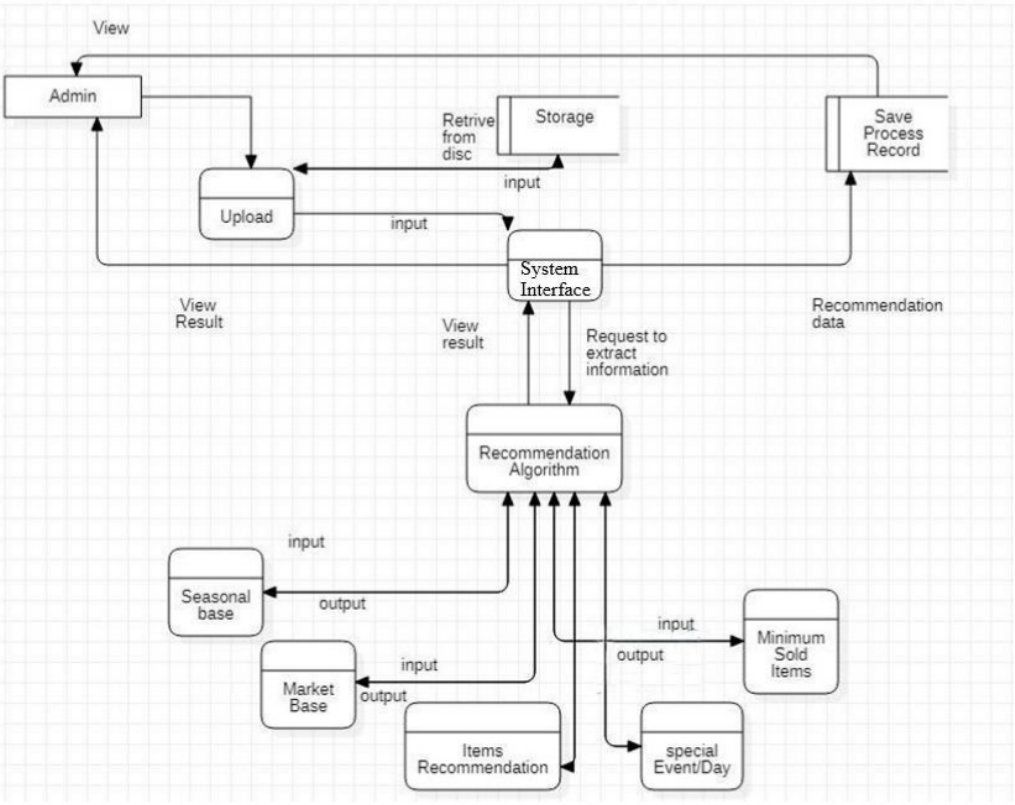

Figure 6: Level 2

4.3 SEQUENCE DIAGRAM

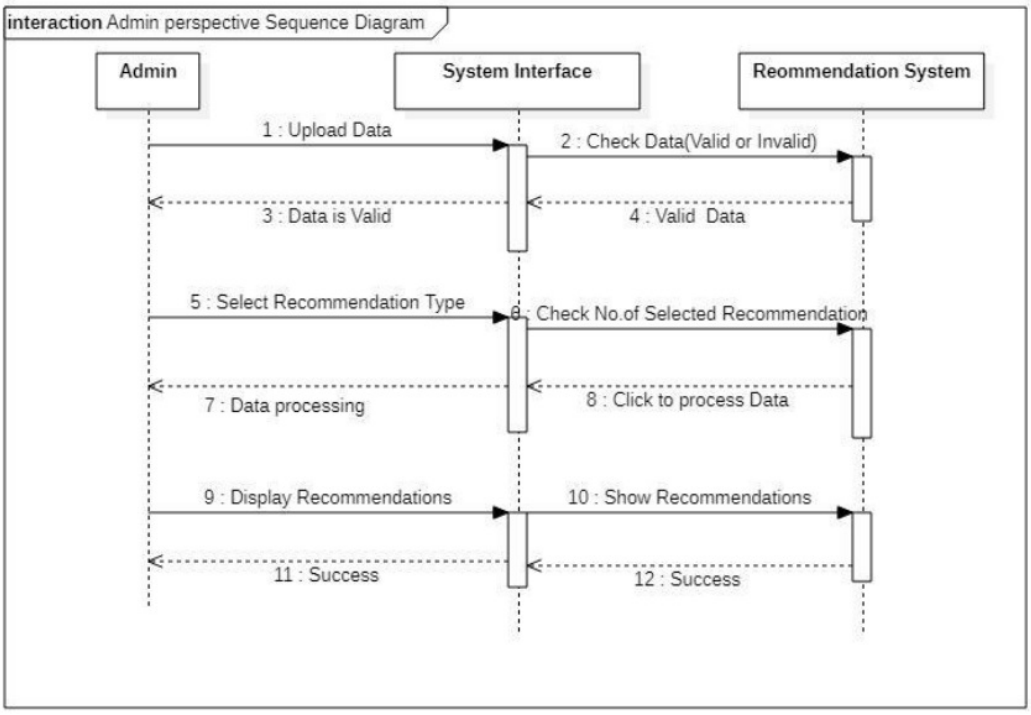

Figure 7: Sequence Diagram

In the Figure 2, Sequence Execution of the functionalities of the 'Admin' of this Recommendation System is illustrated.

Admin can upload the data and the system will check the data is valid or not and if data is valid then Admin can do further process.

After approving the data, the Admin can do further processes (Check the recommendation on uploaded data).

IMPLEMENTATION

5. IMPLEMENTATION

1. IMPLEMENTATION

The Implementation just strategies finishing the activities depicted inside the work plan. Usage of a product venture is an especially awesome crucial it requires the coordination of substantial extent of activities, the heading of a gaggle, the organization of the chose arrangement, and in this way the correspondence to individuals by and large, among various issues. Self-sufficient of whether it is a social assignment to uncover issues and advance towards a computerized or it is an improvement adventure for open movement, there is a certain technique that must be sought after. The going with lines will offer you an introduction into the execution of exercises during a possible programming venture which has been endorsed by a definitive Year Project Committee of COMSATS University, Attock Campus.

5.1.1 METHODOLOGY

Recommendation System for the Purchase Data of the Shopping Arcades is organized using 'The course illustrates'. The Agile model was the most composed improvement technique to influence programming advancements during a methodical way. The Agile Model is simply a period mentioned a once-over of procedures to be performed to encourage an IT framework.[8]

The exercises within the Agile Model are:

5.1.2 SYSTEM ANALYSIS

This progression suggests the social gathering of the system necessity, with the objective of choosing how this essential will be fused into the whole working programming. Expansive correspondence between the customer and along these lines the engineer is fundamental. In System Analysis Feasibility Studies are in like manner passed on to ensure the advancement of a definitive item.

5.1.3 SYSTEM DESIGN

When the need has been accumulated and separated into modules, it's imperative to recognize personally how the system will be created to play out the basic tasks. Even more expressly, the structure arrangement stage is trotted round the data about the system (what is that the last motivation behind the system), the product improvement (by what strategy will the

apparatus be assembled) and in this manner the interface structure and coding (what will the framework look like?)

5.1.4 CODING

Differentiated coding plans were utilized to code the structure's hub. CSS and PyQt apparatuses are utilized to make front association as indicated by customer constrained conditions. PyQt is an anomalous state Python Web structure used to ensure speedy advancement and clean, reasonable arrangement. It is open source and free.

5.2 TOOLS AND TECHNOLOGY

For the development of our project we use following code and here is the brief introduction and advantages of tools we used:

5.2.1 JUPYTER NOTEBOOK

Jupyter Notebook imagines a reality where we utilized PYTHON programming language to actualize information science to our Recommendation system and transform it into a helpful item on the efficient size of programming improvement morals.

Jupyter Notebook Undertaking programming improvement regardless of the size, effectively scaling from a solitary client on one PC to many machines. No cerebral pains, no IT bad dreams.

5.2.2 PYQT 5

PyQt is a Python binding for Qt, which is a collection of C ++ libraries and development tools that include Graphical User Interfaces (GUI) stand-alone speaker, as well as network, threads, standard expressions, SQL data, SVG , OpenGL, XML, and many other powerful features. It helps us to make interactive GUI for our Python project. We have used PyQt version 5.

5.3 FLOW OF WORK

Following is our Project Flow of Work:

5.3.1 DATA GATHERING

First, we needed a dataset for our project, for that we visited a lot of shopping malls and we asked for their shopping mall purchased items dataset. In gathering that data set we have faced a lot of problems:

5.3.2 PROBLEMS IN DATA GTHERING

Following are the problems which we faced during data gathering:

- **Finding General Data**

During Gathering of Data, we did not able to find generic dataset and no one Owner of Shopping Mall is agreed to provide us his/her Purchased Data. Some of owners said that is their personal information that is why they cannot share this.

- **Publicly Available Dataset**

When we could not find any datasets from the shopping malls, we finally had to go for the datasets available online. And we finally got dataset from '**Kaggle'.**

- **Understanding the Dataset**

The main problem after collecting data is to understand the format of the dataset. How many columns are there, what kind of information is given, what information we want to extract from this dataset.

5.4 PRE-PROCESSING

Pre-processing steps are following:

5.4.1 REMOVING UN-NECESSARY ROWS AND COLUMNS

First, we have checked the structure and information given in dataset. then we have removed extra rows which have no data (empty rows). Then we have checked that in description column there are some empty rows (NaN Values) we replace that empty slot with 'XX'. When we delete some extra or empty rows from dataset, we see that the index of rows is not arranged so, we used .reset_index(drop=true) so that we get index reset, (drop = true) means to reset index on same column, no need to add new column. After that we have stored our modified data into new csv file.

5.4.2 ASSIGN UNIQUE CODE TO DESCRIPTION ITEMS

In description there are items which have too large length example ("WHITE HANGING HEART T-LIGHT HOLDER", "FELTCRAFT PRINCESS CHARLOTTE DOLL") and it is not possible to write 6 or more items in same row, so we gave them a unique code to that items. for the sake of giving unique code to the items we take first letter of the items and make a single code. Then we make a dictionary (dic) and put these unique codes in that dic and put original description items names in a new list (des_lis).

- **Problems Occurs**

When we converted that names into code, we checked total length of original names and length of des_lis items then we see there is a lot of difference in length.

- **Problem 1**

we checked that items name in original list is also present in des_lis or not then we come to know that most of the items name is not present in that des_lis then, We have checked this manually for example we have founded a name that is actually not present in dictionary values but we see that item code is present in dictionary code then we come to know that there are some words which are starting with same alphabets and our code assigned them a single unique code e.g. (Apple Orange, Acid Office) code assigned them a single unique value rather than assigning 2 different values.

- Resolving Problem 1

We have modified description items code by picking first and last alphabet of single word. After modifying codes then we again verified that now again length of original items and items in dictionary list is not equal or not.

- Problem 2

After Picking first and last alphabet of single word the problem remains the same because some words has same first and last alphabet.

- Resolving Problem 2

We assign unique number to every word so, after this our problem is resolved and our Description attribute look like this. After this we store this new modified dataset into new csv file.

```
dff = pd.read_csv('New_with_Code.csv')
dff.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WEHGHTTTHR1	6	12/1/2010 8:26	2.55	17850	United Kingdom
1	536365	71053	WEMLLN2	6	12/1/2010 8:26	3.39	17850	United Kingdom
2	536365	84406B	CMCDHSCTHR3	8	12/1/2010 8:26	2.75	17850	United Kingdom
3	536365	84029G	KDUNFGHTWRBE4	6	12/1/2010 8:26	3.39	17850	United Kingdom

Figure 8: Assign Unique code to Description Items

5.4.3 ASSIGN UNIQUE CUSTOMER ID TO EMPTY SLOTS

After making description items codes then we checked that is there any missing data in dataset? we see there are a lot of customers ids are missing. so, we selected these slots and got there mean and assign that mean as id to all the missing customer ids.

```
meanz = int(df.CustomerID.mean())
meanz
```

```
15287
```

```
df['CustomerID'] = df['CustomerID'].fillna(meanz)
```

Figure 9: Assign Unique Customer Id in Empty Slots

5.4.4 MAKING TRANSACTIONAL DATASET

In transactional dataset there is a transaction id and transactional items against that id. for example.

TID	Items in the Basket
1	espresso, sugar, newspaper
2	espresso, sugar, cola
3	espresso, sugar
4	cappuccino, cigarettes
5	cappuccino, sugar
6	cappuccino, sugar, sweets
7	decaf, sugar, chewing_gums
8	decaf, soda, vinegar
9	decaf, sugar, cigarettes

Figure 10: Making Transactional Dataset

Now we need to make a transactional id which is always unique. If we make customer id as a transactional id then we see that a customer who comes one time and complete his/her transaction and then he comes again at same days after one minute then its id remain same but the Invoice Number will be changed on next transaction every time so, we come to know that invoice number is always unique so we make it as transactional id.

First, we find unique ids from the attribute InvoiceNo then we put the description, country and Invoice_date in separate lists and make new dataframe then Put items description country and invoice data against unique Invoice_number.

Transaction_No	Items	Country	Date
536365	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,KDUNFGHTWRBE4,RDWYHEWEH.5,ST77BANGBS6,GSSRFDTTHR7	United Kingdom	12/01/2010 08:26
536366	HDWRUNJK8,HDWRRDPADT9	United Kingdom	12/01/2010 08:28
536367	ADCRBDOT10,PSPEBM11,PSPEKN12,FTPSCEDL13,IYKDMGCY14,BXOF66ADCRTS15,BXOFVEJWBS16,BXOFVEATBS17,HEBGBKWD18,LEBGBKWD19,REBXWHMLHT20,DTNWED21	United Kingdom	12/01/2010 08:34
536368	JMMGSTWHJS22,RDCTRKPSFN23,YWCTRKPSFN24,BECTRKPSFN25	United Kingdom	12/01/2010 08:34
536369	BHBGBKWD26	United Kingdom	12/01/2010 08:35
536370	AMCKBEPK27,AMCKBERD28,AMCKBEGN29,PAADBSSRST30,SSGTTE31,JEPLGE32,VEHSADTSCDGE33,S2RDRTTATS34,RDSKBSSTO4WD35,SYLHBX36,LHBXILELN37,CSPELHBX38,CEBGDYGLDA	France	12/01/2010 08:45
536371	PRCNKTSSCS47	United Kingdom	12/01/2010 09:00
536372	HDWRRDPADT9,HDWRUNJK8	United Kingdom	12/01/2010 09:01
536373	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,ENPLRD48,ROCEMSAD49,SETEPTMG50,VEBDDKMEMG51,VEBDLEMG52,WD22DRCTWEFH53,WDS3CTATWEFH54,WNPEFEWEFH55,WNFEAEWE	United Kingdom	12/01/2010 09:02
536374	VNSGBXLE57	United Kingdom	12/01/2010 09:09
536375	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,ENPLRD48,ROCEMSAD49,SETEPTMG50,VEBDDKMEMG51,VEBDLEMG52,WD22DRCTWEFH53,WDS3CTATWEFH54,WNPEFEWEFH55,WNFEAEWE	United Kingdom	12/01/2010 09:32
536376	HTWRBETAADSY58,RDHGHTTTHR59	United Kingdom	12/01/2010 09:32
536377	HDWRRDPADT9,HDWRUNJK8	United Kingdom	12/01/2010 09:34

Figure 11: Transactional Dataset

5.4.5 ADDING DATE COLUMN

we see that our user just enter date and get result against that specific data so, we do not need time that is why needed to remove time from date column. And we make a new column for date to improve our performance when user enter date our system takes date from date column rather than again and again splitting time from date column it takes too much time and resourced.

```
df_TR['DATE']  = [i.split(' ')[0] for i in df_TR.Date ]
```

	Transaction_No	Items	Country	Date	DATE
0	536365	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,KDUNFGHTWRBE4,...	United Kingdom	12/1/2010 8:26	12/1/2010
1	536366	HDWRUNJK8,HDWRRDPADT9	United Kingdom	12/1/2010 8:28	12/1/2010
2	536367	ADCRBDOT10,PSPEBM11,PSPEKN12,FTPSCEDL13,IYKDMG...	United Kingdom	12/1/2010 8:34	12/1/2010
3	536368	JMMGSTWHJS22,RDCTRKPSFN23,YWCTRKPSFN24,BECTRKP...	United Kingdom	12/1/2010 8:34	12/1/2010
4	536369	BHBGBKWD26	United Kingdom	12/1/2010 8:35	12/1/2010

Figure 12: Adding Date Column

5.4.6 ADDING MONTH COLUMN

For Month Wise and Date Wise Recommendation we need Month and year from user so we make another column and put month and year in that column to improve our performance when user enter Month and Year our system takes date from M_Y (Month and Year) column rather than again and again splitting Month and Year from date column it takes too much time and resourced.

```
df_TR['M_Y']  = ['/'.join(i.split(' ')[0].split('/')[::2]) for i in df_TR.Date ]
```

	Transaction_No	Items	Country	Date	DATE	M_Y
0	536365	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,KDUNFGHTWRBE4,...	United Kingdom	12/1/2010 8:26	12/1/2010	12/2010
1	536366	HDWRUNJK8,HDWRRDPADT9	United Kingdom	12/1/2010 8:28	12/1/2010	12/2010
2	536367	ADCRBDOT10,PSPEBM11,PSPEKN12,FTPSCEDL13,IYKDMG...	United Kingdom	12/1/2010 8:34	12/1/2010	12/2010
3	536368	JMMGSTWHJS22,RDCTRKPSFN23,YWCTRKPSFN24,BECTRKP...	United Kingdom	12/1/2010 8:34	12/1/2010	12/2010
4	536369	BHBGBKWD26	United Kingdom	12/1/2010 8:35	12/1/2010	12/2010
...	...	...	...	...	...	...

Figure 13: Adding Month and Year Column

5.4.7 ADDING MONTH NAME COLUMN

We have created a dictionary and entered month wise names for mapping. We read the date from date column and replace these names with months and make a new Attribute named Month_Name because this is helpful in seasonal base module to direct check through month names like from which month summer is started and from which winter is started, this method can save a lot of resource and enhance our system performance.

```
dic_month_num_alph = {1:'Jan', 2:'Feb', 3:'March', 4:'April', 5:'May', 6:'June', 7:'July', 8:'Aug', 9:'Sep', 10:'Nov', 11:'Oc
```

```
df_TR['Month_Name'] = [dic_month_num_alph[int( dt.split('/')[0] )] for dt in df_TR.M_Y]
```

```
df_TR[df_TR.Month_Name == 'May']
```

	Transaction_No	Items	Country	Date	DATE	M_Y	Month_Name
7149	551515	RDTLLDNTLT40,ADCRMICS265,AMCKBEOE208,TLCDWTKPC...	United Kingdom	5/1/2011 10:51	5/1/2011	5/2011	May
7150	551516	VERDTEMG2421,DYMDSEMG1438,PGSDTAMG1029,NRTEVEF...	United Kingdom	5/1/2011 11:10	5/1/2011	5/2011	May
7151	551517	STOF99HTSDBS253,DTVELSDN3039,BNCESYST1456,JNED...	United Kingdom	5/1/2011 11:14	5/1/2011	5/2011	May
7152	551518	TEHKAEIYRE2990,BEDRWLCK161,RYCD33TR534,WNPEFEW...	United Kingdom	5/1/2011 11:36	5/1/2011	5/2011	May
7153	551519	TLMGCY1037,DTWETOORHE2033,DTHESTHEBE1707,DTAL8...	United Kingdom	5/1/2011 11:50	5/1/2011	5/2011	May

Figure 14:Adding Month Name Column

5.4.8 ADDING SEASON COLUMN

We have added a new column named as season_name, put a loop on Month_Name column and read the month names if a row have November December January then set them as winter in season name column, if a row have May June July then set them as summer in season name column when user choose winter season our system directly get value from Season column rather than getting value from date and then splitting values then compare with months and replace it with month name and then give season name, it take too much time and resources so we make separate column for season to improve our system performance.

```
new = []
for i in df_TR.Month_Name:
    if i =='Nov' or i == 'Dec' or i == 'Jan':
        new.append('Winter')
    elif i =='May' or i == 'Jun' or i == 'July':
        new.append('Summer')
    else:
        new.append(i)
```

```
df_TR['Season_name'] = new
```

```
df_TR.head()
```

	Transaction_No	Items	Country	Date	DATE	M_Y	Month_Name	Season_name
0	536365	WEHGHTTTHR1,WEMLLN2,CMCDHSCTHR3,KDUNFGHTWRBE4,...	United Kingdom	12/1/2010 8:26	12/1/2010	12/2010	Dec	Winter
1	536366	HDWRUNJK8,HDWRRDPADT9	United Kingdom	12/1/2010 8:28	12/1/2010	12/2010	Dec	Winter
2	536367	ADCRBDOT10,PSPEBM11,PSPEKN12,FTPSCEDL13,IYKDMG...	United Kingdom	12/1/2010 8:34	12/1/2010	12/2010	Dec	Winter

Figure 15: Adding Season Column

5.5 RECOMMENDATION IMPLEMENTATION

5.5.1 MONTH WISE RECOMMENDATION

User enters a date for Month wise Recommendation, our system goes to that dataframe and search column according to that date and show results. We put loop on items Attribute and split items on base of space(‘ ’) and make new list, then we use counter function which help us to calculate which item occurs in list and how many times and also give unique items (for example may be apple sold 20 time in different rows then it give us apple 1 time and also tell us that how many times apple is sold).

Then we need items in sorting order, it is difficult to sort a dictionary so we convert dictionary items into list and apply sort function, we sort item on base of first index tuple(sort on basis of numbers), we need highest value first so we put (reverse= true).

Now we need a dataframe to show this result in table form, so we run a loop on 'sessional_lis_M' and put '0' index value in one list and '1' index values in other list and put these value in dataframe.

```
sessional_lis_M = c.Counter([ j for i in df_Sessional_M.Items for j in i.split(',')])
sessional_lis_result_M = sorted(list(sessional_lis_M.items()),key=lambda x: x[1],reverse=True)

code_sess_lis_M = [i[0] for i in sessional_lis_result_M]
code_sess_num_lis_M = [i[1] for i in sessional_lis_result_M]

df_sessional_itemCode_M = pd.DataFrame(columns=['Items','count'])
df_sessional_itemCode_M['Items'] = code_sess_lis_M
df_sessional_itemCode_M['count'] = code_sess_num_lis_M
```

```
df_sessional_itemCode_M.loc[:10]
```

	Items	count
0	WEHGHTTTHR1	213
1	RYCD33TR534	154
2	HDWRBADN422	144
3	PRCNKT5SCS47	143
4	SEDGHTWRBE221	132
5	CEHTWRBE219	124
6	JMMGSTPD79	108
7	PRCNKTVECS169	108
8	HTWRBEBA275	108

Figure 16:Month wise Recommendation

5.5.2 DATE WISE RECOMMENDATION

User enters a date for Date Wise Recommendation, our system goes to that dataframe and search column according to that date and show results. We put loop on items Attribute and split items on base of space(' ') and make new list, then we use counter function which help us to calculate which item occurs in list and how many times and also give unique items.

```
sessional_lis = c.Counter([ j for i in df_Sessional.Items for j in i.split(',')])
```

Figure 17:Date wise Recommendation

Then we need items in sorting order, it is difficult to sort a dictionary so we convert dictionary items into list and apply sort function, we sort item on base of first index tuple(sort on basis of numbers), we need highest value first so we put (reverse= true).

```
sorted(list(dc.items()),key=lambda x:x[1],reverse=True)
```

```
[('WEHGHTTTHR1', 3),
 ('WEMLLN2', 3),
 ('CMCDHSCTHR3', 3),
 ('KDUNFGHTWRBE4', 3),
 ('RDWYHEWEH.5', 3),
 ('ST77BANGBS6', 3),
 ('GSSRFDTTHR7', 3),
 ('HDWRUNJK8', 3),
 ('HDWRRDPADT9', 3),
 ('ENPLRD48', 2),
 ('ROCEMSAD49', 2),
 ('SETEPTMG50', 2),
 ('VEBDDKMEMG51', 2),
 ('VEBDLEMG52', 2),
 ('WD22DRCTWEFH53', 2),
 ('WDS3CTATWEFH54', 2),
```

Figure 18: Date wise Recommendation

Now we need a dataframe to show this result in table form, so we run a loop on 'sessional_list' and put '0' index value in one list and '1' index values in other list and put these value in dataframe.

```
code_sess_lis = [i[0] for i in sessional_lis_result]
code_sess_num_lis = [i[1] for i in sessional_lis_result]
```

```
df_sessional_itemCode = pd.DataFrame(columns=['Items','count'])
df_sessional_itemCode['Items'] = code_sess_lis
df_sessional_itemCode['count'] = code_sess_num_lis
```

```
df_sessional_itemCode
```

	Items	count
0	HDWRSYDGDN198	16
1	WEHGHTTTHR1	15
2	HDWROLDN199	13
3	HDWRRDRT200	13
4	RDWYHEWEH.5	12
...	...	...
949	S4GNREDRCE944	1
950	GDPTPRBG945	1
951	WNHYBYGD946	1

5.5.3 SEASONAL RECOMMENDATION

- Summer Season

User Clicks Summer Season for Seasonal Recommendation, our system goes to that dataframe and search column according to that Season and show results. We put loop on items

Attribute and split items on base of space(‘ ’) and make new list, then we use counter function which help us to calculate which item occurs in list and how many times and also give unique items.

Then we need items in sorting order, it is difficult to sort a dictionary so we convert dictionary items into list and apply sort function, we sort item on base of first index tuple(sort on basis of numbers), we need highest value first so we put (reverse= true).

Now we need a dataframe to show this result in table form, so we run a loop on ‘summer_lis_result_M’ and put ‘0’ index value in one list and ‘1’ index values in other list and put these value in dataframe.

```
summer_df = df_TR[df_TR.Season_name == 'Summer']

summer_lis_M = c.Counter([ j for i in summer_df.Items for j in i.split(',')])
summer_lis_result_M = sorted(list(summer_lis_M.items()),key=lambda x: x[1],reverse=True)

code_summer_lis_M = [i[0] for i in summer_lis_result_M]
code_summer_num_lis_M = [i[1] for i in summer_lis_result_M]


df_summer_itemCode_M = pd.DataFrame(columns=['Items','count'])
df_summer_itemCode_M['Items'] = code_summer_lis_M
df_summer_itemCode_M['count'] = code_summer_num_lis_M
```

Figure 19: Summer Season

- Winter Season

User Clicks Winter Season for Seasonal Recommendation, our system goes to that dataframe and search column according to that Season and show results. We put loop on items Attribute and split items on base of space(‘ ’) and make new list, then we use counter function which help us to calculate which item occurs in list and how many times and also give unique items.

Then we need items in sorting order, it is difficult to sort a dictionary so we convert dictionary items into list and apply sort function, we sort item on base of first index tuple(sort on basis of numbers), we need highest value first so we put (reverse= true).

Now we need a dataframe to show this result in table form, so we run a loop on ‘Winter_lis_result_M’ and put ‘0’ index value in one list and ‘1’ index values in other list and put these value in dataframe.

```
winter_df = df_TR[df_TR.Season_name == 'Winter']

winter_lis_M = c.Counter([ j for i in winter_df.Items for j in i.split(',')])
winter_lis_result_M = sorted(list(winter_lis_M.items()),key=lambda x: x[1],reverse=True)

code_winter_lis_M = [i[0] for i in winter_lis_result_M]
code_winter_num_lis_M = [i[1] for i in winter_lis_result_M]


df_winter_itemCode_M = pd.DataFrame(columns=['Items','count'])
df_winter_itemCode_M['Items'] = code_winter_lis_M
df_winter_itemCode_M['count'] = code_winter_num_lis_M
```

Figure 20: Winter Season

5.6 APRIORI ALGORITHM IMPLEMENTATION

5.6.1 MAKING BINARY DATASET

We have changed our dataset structure to convert it into binary dataset because, Apriori algorithm and related libraries work good and efficient on binary dataset and to get Association Rules of Apriori Algorithm.

```
Columns = list(np.unique([ j for i in df.Items for j in i.split(',')]))
Columns = ['Transaction_No']+Columns
len(Columns)
```

```
3920
```

```
dfx = pd.DataFrame(index=np.arange(0, df.shape[0]+1),columns=Columns)
dfx.head(2)
```

Transaction_No	10CRSYPN307	11HR	1228	12CDPYBS1147	12DYPSINWDBX376	12EGHEPDWD2414	12HGESHDPD3862	12IYREPGPESS15
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N

2 rows × 3920 columns

Figure 21: Binary Dataset

we have picked transaction number and item from table sequence wise and replaced 1 if Transaction number and item is present, if item is not present against that transaction number then replaced with 0. And stored this modified dataset in binary_db_1.csv.

```
for row,x,Item_lis in zip(range(0,df.shape[0]+1),df.Transaction_No,df.Items):
    dfx.iloc[row]['Transaction_No'] = x

    #print(Item_lis.split(','))
    for item_ in Item_lis.split(','):
            I
        dfx.iloc[row][item_] = 1
```

```
dfx.head()
```

	Transaction_No	Unnamed: 1	10CRSYPN307	11HR	1228	12CDPYBS1147	12DYPSINWDBX376	12EGHEPDWD2414	12HGESHDPD3862	12IYREPGF
0	536365	0	0	0	0	0	0	0	0	
1	536366	0	0	0	0	0	0	0	0	
2	536367	0	0	0	0	0	0	0	0	
3	536368	0	0	0	0	0	0	0	0	
4	536369	0	0	0	0	0	0	0	0	

5 rows × 3920 columns

Figure 22: Binary Dataset

5.6.2 APPLYING APRIORI ALGORITHM

Applied Apriori Algorithm on binary dataset. to calculate the support for items we check in row for single item that how many time this item sold and get sum of whole row against that item and store it in dictionary then make a dataframe make single column named Support and pass the values before column.

```
dic = dict()
for i in df.columns[0:]:
    dic[i]  = df[i].sum()
```

```
d_d = pd.DataFrame((list(dic.values())) , columns= ["Support"], index=[list(dic.keys())])
```

```
d_d
```

	Support
10CRSYPN307	249
11HR	58
1228	127
12CDPYBS1147	140
12DYPSINWDBX376	62

Figure 23: Applying Apriori Algorithm

We have passed whole binary dataframe to Apriori algorithm then set minimum support and set column name = true, means that when our algo give us support for items then original column name will be return.

```
frequent_itemsets_ = apriori(dfx, min_support=0.02, use_colnames=True)
frequent_itemsets_['length'] = frequent_itemsets_['itemsets'].apply(lambda x: len(x))
frequent_itemsets_
```

	support	itemsets	length
0	0.031594	(60TEFYCECS73)	1
1	0.032793	(66RSRCCM375)	1
2	0.032194	(72STFYCECS293)	1
3	0.056589	(ADCRBDOT10)	1
4	0.027994	(AESRTAGSED1160)	1
...	...	...	...
168	0.026795	(JMMGSTPD79, JMMGSTWHJS22)	2
169	0.023395	(STOF33CETSPYDN2166, JMMGSTWHJS22)	2
170	0.031394	(WEHGHTTTHR1, RDHGHTTTHR59)	2
171	0.034193	(STOF33CETSPYDN2166, STOF66SETSPYDN2165)	2
172	0.026195	(WNPEFEWEFH55, WNFEAEWE56)	2

173 rows × 3 columns

Figure 24: Applying Apriori Algorithm

5.6.3 HIGHEST SOLD ITEMS

We have calculated item support, based on support we can find highest support item, means that highest sold item.

```
d_d_top = d_d[d_d.Support == sorted(d_d.Support)[-1]]
d_d_top
```

	Support	items
WEHGHTTTHR1	2013	(WEHGHTTTHR1,)

Figure 25: Highest Sold Item

5.6.4 MINIMUM SOLD ITEMS

Based on support we can find lowest support item, means that highest sold item.

```
d_d_low = d_d[d_d.Support == sorted(d_d.Support)[0]]
d_d_low
```

	Support	items
16PCCYSTPYDN3194	1	(16PCCYSTPYDN3194,)
33T	1	(33T,)
55SDGSNEAT3833	1	(55SDGSNEAT3833,)
72CECSVECS1742	1	(72CECSVECS1742,)
ACJLSK2262	1	(ACJLSK2262,)
...	...	...
WPBERNFT1767	1	(WPBERNFT1767,)
WPPKFK1308	1	(WPPKFK1308,)
WYHTSKGEATSG2189	1	(WYHTSKGEATSG2189,)
ZCPTPTHR3160	1	(ZCPTPTHR3160,)
ZCSRTTHR3199	1	(ZCSRTTHR3199,)

212 rows × 2 columns

Figure 26: Lowest Sold Items

5.6.5 ASSOCIATION RULES

Association rules from Apriori Algorithm.

```
rules_ = association_rules(frequent_itemsets_, metric="confidence", min_threshold=0.01)
rules_
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(AMCKBERD28)	(AMCKBEGN29)	0.041792	0.039592	0.027395	0.655502	16.556401	0.025740	2.787851
1	(AMCKBEGN29)	(AMCKBERD28)	0.039592	0.041792	0.027395	0.691919	16.556401	0.025740	3.110250
2	(WEHGHTTTHR1)	(CRPKHGHT264)	0.116977	0.028994	0.022795	0.194872	6.721061	0.019404	1.206026
3	(CRPKHGHT264)	(WEHGHTTTHR1)	0.028994	0.116977	0.022795	0.786207	6.721061	0.019404	4.130271
4	(RSRYTPADSR623)	(GNRYTPADSR628)	0.037992	0.035793	0.026395	0.694737	19.409938	0.025035	3.158610
5	(GNRYTPADSR628)	(RSRYTPADSR623)	0.035793	0.037992	0.026395	0.737430	19.409938	0.025035	3.663816
6	(HTOFWRSL128)	(HTOFWRLE129)	0.067187	0.056989	0.033593	0.500000	8.773684	0.029764	1.886023
7	(HTOFWRLE129)	(HTOFWRSL128)	0.056989	0.067187	0.033593	0.589474	8.773684	0.029764	2.272238
8	(WEHGHTTTHR1)	(HTOFWRLE129)	0.116977	0.056989	0.023595	0.201709	3.539469	0.016929	1.181288
9	(HTOFWRLE129)	(WEHGHTTTHR1)	0.056989	0.116977	0.023595	0.414035	3.539469	0.016929	1.506956
10	(WEHGHTTTHR1)	(HTOFWRSL128)	0.116977	0.067187	0.022595	0.193162	2.875015	0.014736	1.156135
11	(HTOFWRSL128)	(WEHGHTTTHR1)	0.067187	0.116977	0.022595	0.336310	2.875015	0.014736	1.330475

Figure 27: Association Rules

EVALUATION
AND
TESTING

6. EVALUATION AND TESTING

6.1 EVALUATION AND TESTING

The Software System is tried to assess the functionalities as per request and needs of a client or with a goal to discover whether the created programming met the predetermined necessities or not and to recognize the deformities/mistakes/bugs to guarantee the nature of our item. Testing was completed by us in the referenced kinds that are:

6.1.1 TEST CASES

Test cases helped use to investigate the working of this software product as indicated by the view of a client who is an ordinary user of Recommendation System.

Test Case Name: Data Uploading

Project Name: Recommendation System for the Purchase Data of the Shopping Arcades

Pre-Condition Application Opening

Steps 3

No#	Steps	Expected Results	Actual Results	Pass/Fail
1	Running Application without initializing Data.	Error will generate and cannot get Recommendations.	The error was generated.	Pass
2	Uploaded the invalid data.	The error will generate that the data is invalid.	The error was generated.	Pass

3	Proceed Application by Initializing Data.	System will proceed the data.	System proceeded the data.	Pass

Post Condition: Selecting Recommendation type				
Test Case Name: Selecting Recommendation Type **Project Name: Recommendation System for the Purchase Data of the Shopping Arcades** **Pre-Condition: Data is Initialized** **Steps: 1**				
No#	**Steps**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Select desired Recommendation.	The detail page of that specific recommendation will display.	The detailed page of that specific product was displayed.	Pass
Post Condition: Selecting Recommendation				
Test Case Name: Recommendations **Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.** **Pre-Condition: Data is Initialized** **Steps: 2**				

No#	Steps	Expected Results	Actual Results	Pass/Fail
1	Select Month wise recommendation.	Recommendation system will show recommendations according to Month wise.	Recommendation system shown recommendations according to Month wise.	Pass
2	Select Date wise Recommendation.	Recommendation system will show recommendations according to Date wise.	Recommendation system will show recommendations according to Date wise.	Pass

Post Condition: Selecting Recommendation

Test Case Name: Recommendations

Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.

Pre-Condition: Data is Initialized

Steps: 2

No#	Steps	Expected Results	Actual Results	Pass/Fail
1	Select Less Demanding Item's recommendation.	The detailed page of Less Demanding item's	The detailed page of Less demanding Item's was displayed.	Pass

		recommendation will display.		
2	Select High Demanding Item's recommendation.	The detail page of High Demanding item's recommendation will display.	The detailed page of High demanding Item's was displayed.	Pass

Post Condition: Selecting Recommendation				
Test Case Name: Recommendations **Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.** **Pre-Condition: Data is Initialized** **Steps: 1**				
No#	**Steps**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Type Date for Special Events Recommendation.	The detailed page of Special Events recommendation will display.	The detailed page of Special Events was displayed.	Pass

Post Condition: Selecting Recommendation

Test Case Name: Recommendations

Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.

Pre-Condition: Data is Initialized

Steps: 1

No#	Steps	Expected Results	Actual Results	Pass/Fail
1	Select Items for Sale Button.	The detailed page of Items for Sale Recommendation will display.	The detailed page of Items for Sale was displayed.	Pass

Post Condition: Selecting Recommendation

Test Case Name: Recommendations

Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.

Pre-Condition: Data is Initialized

Steps: 2

No#	Steps	Expected Results	Actual Results	Pass/Fail
1	Select Summer season Button.	The detailed page of Summer	The detailed page of Summer	Pass

		Season recommendation will display.	Season was displayed.	
2	Select Winter Season Button.	The detail page of Winter Season recommendation will display.	The detailed page of Winter Season was displayed.	Pass
Post Condition: Selecting Recommendation				
Test Case Name: Recommendations **Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.** **Pre-Condition: Data is Initialized** **Steps: 3**				
No#	**Steps**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Select Most Suitable Group Suggestions Button.	The detailed page of Suitable Group Suggestions recommendation will display.	The detailed page of Suitable Group Suggestions was displayed.	Pass

2	Select Rule for Person Button.	The detail page of Rule for Person recommendation will display.	The detailed page of Rules for Person was displayed.	Pass
3	Select Rules Button.	The detail page of Rules recommendation will display.	The detailed page of Rules was displayed.	Pass
Post Condition: Selecting Recommendation				
Test Case Name: Recommendations **Project Name: Recommendation System for the Purchased Data of the Shopping Arcades.** **Pre-Condition: Data is Initialized** **Steps: 1**				
No#	**Steps**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Select Top soled Items Button.	The detailed page of Top Sold Items recommendation will display.	The detailed page of Top Sold Items was displayed.	Pass

6.2 UNIT TESTING

Each single module of this Recommendation shopping System was tried during the improvement procedure to affirm its working legitimate filling in according to required. Testing was performed on each individual module. The designer by and by tried each line of code to guarantee the productivity and toughness of the code.

6.3 FUNCTIONAL TESTING

Functional testing was performed on the following key modules of the Recommendation System for the Purchase data of the Shopping Arcades:

- **Uploading Data**
- **Recommendations**
- **Market Need**
- **Special Events**
- **Minimum Sold Items**
- **Seasonal Recommendation**
- **Rules for Recommendation**

Functionalities were tried by means of following testing methods and client's point of view was remembered. Testing as indicated by client's point of view consistently assists with improving the item and make it progressively effective and intuitive.

6.3.1 BLACK BOX TESTING

Black Box Testing was performed by clients on outer pieces of the framework. The client had no clue about the end code. The client utilized an outer portrayal of the Recommendation System, coordinated them with the referenced necessities organized the aftereffects of the testing.

The upsides of black box testing are, As the testing was performed by a client and designer took part in it that is the reason the testing is adjusted. For a client who needs to support his deal isn't committed to know about the application's code/inside structure and programming information, when all is said in done, isn't required for utilization of this Recommendation System.

The test was done in consideration of different regular users of the Recommendation system.

6.3.2 WHITE BOX TESTING

White Box testing is exceptionally performed by a group experienced engineer to guarantee the specific working of this Recommendation System for the Purchase information of the Shopping Arcades. Software engineers got to the source code for each conceivable escape clause including the dangers of harm for the Recommendation framework while running on the System. Experienced designers tried the product and their comprehension helped us to streamline our code, learn effective programming and it helped us to understand many concealed imperfections. White Box Testing by an accomplished developer is as significant as Black Box testing from a standard client.

CONCLUSION
AND
FUTURE WORK

7. CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

After extensive research and testing, we have come to this conclusion. Of course, buying daily necessities from shopping malls is popular among Pakistanis. To save time and for the convenience of the people, it is important to keep the items in order. Items that sell well at the same time should be arranged according to the needs of the people. Most people prefer large shopping malls because most items are necessarily available in one place. And with the availability of items, if the items are arranged in a better way, it also makes it easier for people and more items are likely to be sold. After this investigation, we understand that shopping malls have a huge role to play in Pakistan. And this industry has a lot of potential for growth as the number of shopping malls and the needs of the people are increasing day by day.

7.1.1 ANDROID APP

In future we are going to dispatch an android application for our framework to widen the utilization of our application. Since the use of applications in turning out to be increasingly more helpful now a days and it additionally expands the transportability and the extensibility of task.

References:

- https://www.geeksforgeeks.org/apriori-algorithm/ [1]
- http://rejoiner.com/resources/amazon-recommendations-secret-selling-online/#:~:text=Amazon%20currently%20uses%20item%2Dto,recommendation%20list%20for%20the%20user. [2]
- https://www.kdnuggets.com/2019/08/order-matters-alibabas-transformer-based-recommender-system.html#:~:text=Ranking%20Candidates&text=The%20retrieval%20step%20used%20at,these%20candidates%20with%20high%20precision. [3]
- https://tech.flipkart.com/e-commerce-recommendations-using-machine-learning-7002526e531a [4]
- https://open.blogs.nytimes.com/2015/08/11/building-the-next-new-york-times-recommendation-engine/ [5]
- Basilico, J. and Hofmann, T., 2004, July. Unifying collaborative and content-based filtering. In *Proceedings of the twenty-first international conference on Machine learning* (p. 9). [6]
- Pazzani, M.J. and Billsus, D., 2007. Content-based recommendation systems. In *The adaptive web* (pp. 325-341). Springer, Berlin, Heidelberg. [7]
- https://www.tutorialspoint.com/sdlc/sdlc_agile_model.htm#:~:text=Agile%20SDLC%20model%20is%20a,builds%20are%20provided%20in%20iterations. [8]

https://www.kaggle.com/datasets

Recommendation System for the Purchase Data of the Shopping Arcades

ORIGINALITY REPORT

17%	3%	1%	17%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Chiang Mai University Student Paper	12%
2	Submitted to University of Wales central institutions Student Paper	1%
3	Submitted to City University Student Paper	<1%
4	Submitted to University of Wolverhampton Student Paper	<1%
5	viitorcloud.com Internet Source	<1%
6	www.javatpoint.com Internet Source	<1%
7	Submitted to University of Wales Swansea Student Paper	<1%
8	Submitted to University of Westminster Student Paper	<1%

9	www.tandfonline.com Internet Source	<1%
10	Joshua M. Willman. "Beginning PyQt", Springer Science and Business Media LLC, 2020 Publication	<1%
11	www.kdnuggets.com Internet Source	<1%
12	Submitted to Middlesex University Student Paper	<1%
13	Submitted to Indian Institute of Management Student Paper	<1%
14	senior.ceng.metu.edu.tr Internet Source	<1%
15	Submitted to Anglia Ruskin University Student Paper	<1%
16	Submitted to Higher Education Commission Pakistan Student Paper	<1%
17	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1%
18	www.theseus.fi Internet Source	<1%
19	Submitted to Bournemouth University	

	Student Paper	<1%
20	scholarcommons.scu.edu Internet Source	<1%
21	art-unwashed.blogspot.com Internet Source	<1%
22	Submitted to Ganpat University Student Paper	<1%
23	Submitted to University of the West Indies Student Paper	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		