Design product placement layout and personalized discount based on Customer Travel Path

A Thesis Submitted for the Degree of Master of Computer Science

R.S.N Dilrukshi

University of Colombo School of Computing

2021

UCSC

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Student Name: R.S.N Dilrukshi

Registration Number:2016/MCS/028

Index Number:16440289

___ ____ (2021/11/29) _

Signature of the Student & Date

This is to certify that this thesis is based on the work of ~~Mr~~. /Ms. RSN Dilrukshi under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by,

Supervisor Name:

H. A. Caldera_____________30-11-2021___________

Signature of the Supervisor & Date

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to all those who provided me the possibility to complete this thesis. I offer my special gratitude to my supervisor Dr. H.A. Caldera, who given me valuable guidance and direction to make this task success.

I would like sincerely to thank all the lectures and the post-graduate department who invested the full effort in achieving the goal.

Last but not the least, I would like to thank you my husband, family and my friends for supporting and encouraging me throughout this thesis and the MSc program.

ABSTRACT

In today's competitive market, understanding its consumers is key to the success of any business. The market contains various consumer subgroups that can be distinguished based on purchasing habits, time spent, product selection, and travel path. To identify the pattern hidden inside these subgroups, real data is needed as it reflects the ordinary behaviour of consumers.

Analysis of the travel path data that consumers make inside the shopping mall enables retailers to understand and predict consumer behaviour, which has become a critical point in effective decision making. Based on the travel path through the proposed methodology, it demonstrates an approach which uses the Frequent Pattern Growth (FP Growth) algorithm in order to improve sales based on personalized discount schemas and an improved store layout.

The RFM (Recency, Frequency, Monitory value) analysis method has been used in order to identify the customer segments based on the dataset of Instacart from the Kaggle website. An FP growth algorithm has been used to identify the frequent locations and frequent products of consumers. An improved version of the supermarket layout has been suggested based on the frequent travel areas of consumers. The findings of this approach can be used by retailers to improve the in-store shopping experience of consumers.

Keywords: FP growth algorithm (FP growth), RFM analysis, personalized discount, shopping layout

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

People have developed new discoveries to make mankind's lifestyle easier and more flexible. Shopping is one of the main investments of energy people spend for their benefit. Shopping includes garments, electrical machines, food items and so forth. The supermarket is rapidly growing by providing facilities for the high class to middle class segments. As technology gets updated with new trends, each of the retailers tries to make their customers' shopping mall experience more and more interesting and easy.

According to industry experts, the typical store requires remodelling ten years after it opens and every six to seven years thereafter to improve its looks, efficiency, and operation. A typical customer decides their next return to the store based on the impression they get from the layout of the store. From the perspective of the retailer, the layout of the customer determines the exposure of the customer to the goods and affects the chance of the item being bought. Except for some intuitive guidelines used by retailers in store layout design (for instance, to locate coffee and sugar together or shampoo and conditioner together), the number of analytical layout design models for retail stores in the literature is limited (Griswold, et al., 2004). In the supermarket layout aspect, retailers need to consider allocating space to the products on shelves. Due to the recent competition in the retailing industry, retailers are striving to improve their revenue in order to run their stores more efficiently (Husen & Lee, 2014). The term competition in retail is the rivalry among retailers who are keen to obtain the same customer. To improve sales and revenue, various analyses are performed by a retailer to determine which different products should be merchandized together based on historic purchasing behaviour. According to research by Tejada, García-Vázquez, & Brena (2014), efficient shelf space allocation management does not only minimize the economic threat of empty product shelves; it can also lead to higher consumer satisfaction, a better consumer relationship, and even more importantly, it can have a significant positive effect on product sales (Tejada, García-Vázquez, & Brena, 2014).

1.1 Motivation

The large amounts of data stored on the server can be used to learn about trends in each specific field of use. Similarly, the transaction data of each retailer can be used to determine the patterns of their buyers and use that result to increase their revenue further. The discounts or promotions are applied based on the items they are purchasing, or else they issue a percentage discount on the total order for any customer. Many of the customers selected a particular supermarket for their purchase. Based on the travel path, giving a customized discount is a good option to build a strong customer relationship. In supermarkets, items are placed in a common layout form where same-category items are placed nearby. The layout of the supermarket can be divided into several areas based on travel frequency. To identify those blocks, they can use tracking and, by analysing those paths, supermarkets can easily remodel the layout of the market to expand their travel paths around the supermarket.

1.2 Statement of the problem

In the current supermarkets, retailers are focusing on growth for their business based on the quality of the product and providing better service to their customers. The layout of the supermarket is much more important in order to maximize the buying power of the customers. If many customers are not visiting some particular shelves, those won’t be able to be identified in the current context. This may require changing the layout of the item placement in order to make the customers go around the market without missing any shelves. In supermarkets, aisle locations play a major role, as they will lead you to identify the pattern of movements inside the supermarkets. Retailers use discounts to attract consumers to their shops. But discounts are given in such a manner that customers get the same amount of discount for the same item irrespective of whether they are regular or not.

In order to make customer relationships more satisfying, the above data can be used to identify frequent travel patterns and, based on that, give a customized discount for particular products. Identified frequent paths can be used to recognize the frequent and non-frequent travel aisles of the customer, and that can be used to update the shop layout accordingly.

1.3 Research Aims and Objectives

1.3.1 Aim

The aim of this project is to design a product placement layout and personalized discount based on customer travel path.

1.3.2 Objectives

The main objective of this project is to develop a product placement layout and suggest a personalized discount based on the customer's travel path inside the supermarket. Using the customer travel path data can help in identifying their purchase behavior, time spent on purchasing, and improving sales by suggesting discounts.

The specific objectives of the project are as listed below

- To motivate customer to buy the products which are located in their traveled path but not bought.
- To identify the most traveled path by customers
- To improve the sales by expanding the travel area of the customer.
- Suggest a product placement layout by analyzing data of customers' travel paths inside the supermarket in order to maximize the sales.

1.4 Scope

The scope of this project is to develop an application that will allow us to give specific discounts to each customer of the supermarket based on their travel path inside the supermarket. To expand customer travel paths, the project will introduce a discount scheme for items that are in their frequent travel path and may or may not be bought. By analyzing the data of the travel paths, an effective shopping layout will be suggested. Frequent traveled paths will be able to be identified through analyzing the traveled path data, and non-frequent sale items will be placed along these paths with personalized discounts for consumers.

1.5 Structure of the Thesis

This document consists of five chapters. Chapter one provides a brief introduction to the project background, motivation, problem domain and project objectives with the aim. In chapter two, it addresses the literature review, results of similar research work, comparing their results and providing an insight into this project. In chapter three, it defined the methodology of the proposed system. Chapter four, outlines the aspects such as evaluation methods and designed experiments with the results obtained. Finally, chapter five outlines the conclusion with the limitation of current findings and work for the future.

CHAPTER 2
LITERATURE REVIEW

Through this chapter, it will analyze the finding of similar projects that align with the primary objective based on layout analysis, frequent pattern mining algorithms, and customer segmentation techniques. This chapter will further give an insight into data mining techniques such as association rule mining and clustering techniques in the domain of shopping.

2.1 Store layout affect for the Sales

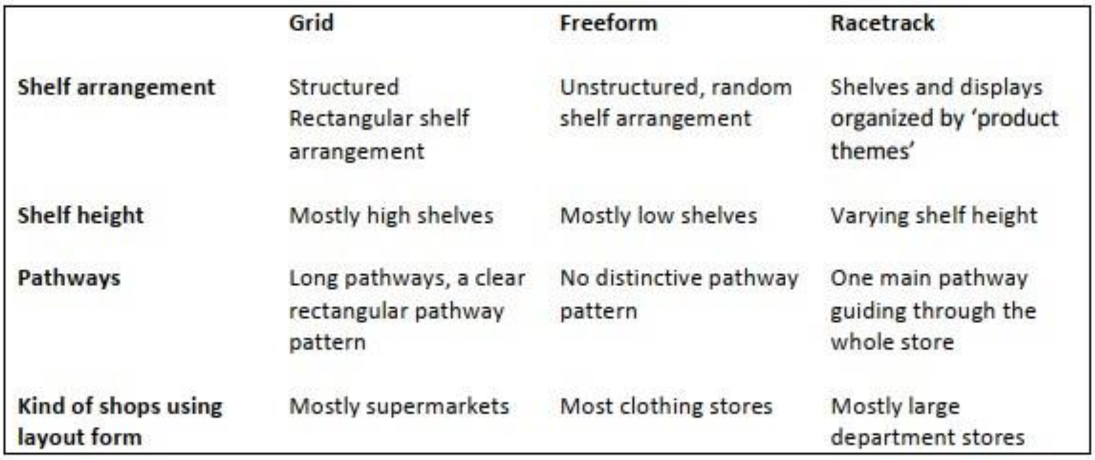
In the layout design, retailers need to consider how the allocation of the product on shelves should be. Through efficient shelf management, leads to better consumer relationships, higher customer satisfaction and also a positive effect on the product sales (Liu, Melara, & Arangarasan, 2007).

As stated in research done by Chandon et al. (2009), in order to diversify the promotion in the market, marketers are moving from traditional out-of-store media advertising to the in-store advertising. They manage shelf management and audience measurement tools as a response to it. According to some studies, the position of brands in the horizontal or vertical display influences the choices and quality of expectations. Many of the choices by consumers for product buying are made inside the store. Instore marketing activities influence consumer behavior at the purchase point, and based on the findings show that up to a certain extent (Chandon, Hutchinson, & Bradlow, 2009). According to the finding, not just the choice given consideration or consideration given the attention past item usage but past brand also increase attention toward the brand. According to the research it suggests that brand not only increase the expected utility of the brand but also increase expected utility of the brand. For the managers who are interested in metrics of point-of-purchase behavior, the results show that behaviors categorized into two groups. They are based on higher order evaluation process or depend on the attention and measures by eye movements. In considering about ‘recall’ it generally defines as about attention. In considering recall more biased toward the highly branded products and for recall it need sufficient amount of elaboration. As an example, people who are much educated recall more brands but they notice less of them on the shelves. As because of that marketer need to measure attention but not just about evaluation and eye tracking (Chandon, Hutchinson, & Bradlow, 2009).

In considering about the store layout design there are two types which are aisle design and shelf design.

2.1.1 Aisle Design

According to Liu et al. (2007) inside a store, personal space is one of the comfort experiences expected by the consumers. This allows consumers to influence retail experiences and also make actual choices in the store. In research by Levav & Zhu (2009) state that the amount of perceived space a consumer has influences the choice the consumer makes inside a store. Consumers who are spatial confinement are tend to make seeking in their purchasing. Through the findings the results in purchase behavior in which consumers tend to choose more products that they can use to carry out their distinctive identity. In a crowded shopping environment people are more likely to focus on prevention, resulting in safety-related product choice (Levav & Zhu, 2009). In present retailing industry three common layout types are use: grid, freeform and racetrack layout (Liu, Melara, & Arangarasan, 2007). The type of the layout selecting have great influence to the image of the store and this image is affect to the behavior of the consumers. Through the layout design depend internal traffic pattern and operational behavior of the store (Lewison, 1996). Consumer satisfaction also depend on the store layout. Figure 2.1 illustrated the characteristics of different aisle designs which stated in research done by Elbers (2016). These all characteristics are not applicable to every store according to Elbers (2016) as because it different based on the industry and the consumers interests.



	Grid	Freeform	Racetrack
Shelf arrangement	Structured Rectangular shelf arrangement	Unstructured, random shelf arrangement	Shelves and displays organized by 'product themes'
Shelf height	Mostly high shelves	Mostly low shelves	Varying shelf height
Pathways	Long pathways, a clear rectangular pathway pattern	No distinctive pathway pattern	One main pathway guiding through the whole store
Kind of shops using layout form	Mostly supermarkets	Most clothing stores	Mostly large department stores

Figure 2.1: Layout design overview (Elbers, 2016)

2.1.2 Shelf Design

The structure of the shelf design has advantages for both retailers and consumer in that the buyers' shopping satisfaction gets increased if the shelf design structure is well. Dreze et al. (1994) mentioned that many of the consumer decisions on purchasing are made prior to visiting the store. In the in-store environment, buyers follow a quick review of the price comparison and product search. For the consumer, in-store experience depends on how the products are presented on the shelves, the number of facings and their brands. For the effectiveness of the shelf design, product placement on the shelf, category arrangement, number of products facing and product adjacencies are affected. For product placement, consider horizontal positioning and vertical positioning (Liu, Melara, & Arangarasan, 2007).

2.1.2.1 Horizontal Positioning

According to the research by Valenzuela et al. (2013), customers consider products that are in the middle of the shelf to be the most popular ones. According to Sorensen's 2003 research, products in the horizontal extremes of shelves attract more consumer attention than products in the middle of the shelf. When products are in the horizontal form, they are easy to reach when the customers reached from the main way. Also, when the shopping carts are placed at a place, it doesn't mean that the products placed at this extreme are given the most face time by consumers. There is insufficient evidence to demonstrate the effects on different horizontal product locations have on consumer behavior and product sales.

2.1.2.2 Vertical Positioning

Horizontal product placement has more effect than the vertical product positioning on shelves (Valenzuela & Raghubir , Center of Orientation: Effect of Vertical and Horizontal Shelf Space Product Position, 2009). Vertical location effects have more impact on sales than the horizontal shelf lengths. The most effective location for product placement is the eye level (Larson, Bradlow, & Fader, 2005).

When the consumer travel through supermarket they may check all the shelves in the mall or some of the shelves or one shelves and leave the market. In considering the path where all the consumers are traveled marketers able to get a good idea how the client's interests are and how the layout can possibly be changed.

2.2 Shopping path Analysis

The research was based on categorizing the path traveled by each shopper using a clustering algorithm and identifying 14 different canonical paths for its customers(Larson, Bradlow, & Fader, 2005). According to the study, Figure 2.2 shows shopper behavior in a supermarket that was tracked through "PathTracker" software by Sorensen Associated, an in-store research firm. In order to find the items, shoppers travel back and forth from one store to another, resulting in a lot of impulse purchases. To increase the number of purchases by customers, grocery stores need to carefully design their layout. In considering Figure 2.2, customers do not consider all the areas of the mall in an equal way. They go through different areas at different speeds. Some of the areas have drawn more attention than other parts of the layout.

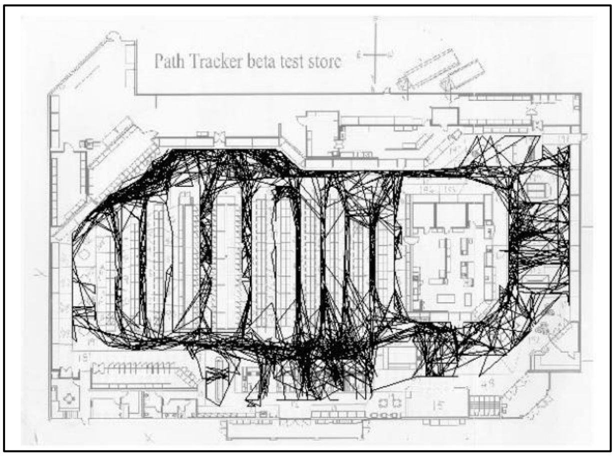


Figure 2.2: 'PathTracker' data from 20 random customers (Larson, Bradlow, & Fader, 2005)

Most areas that get the attention of the customer may include:

- Area at the entrance of the store including the area of display or first shelf that customer face immediately after enter to the store.
- End cap of aisles that visible for people who passed shelf but not enter into the aisle.
- Check out area, that all the customers have to pass for make payments.

According to these designs, instore shelf design applicable to different types of stores. Adapting the product allocation accordingly is affect to the significant of the product sales. The best option depends type of the store and retailers' goal to provide consumer an enjoyable and interesting shopping experience which provide them more benefit than expect.

2.2.1 Pattern Discovery

Pattern discovery from a sequence of data is one of main task in data mining research area. These techniques can be applied to many domains. In knowledge discovery process data mining use different approaches such as classification, clustering and association.

2.2.1.1 Association Rule Mining

Main concept behind approach suggested by Alyoubi in 2020 to extract hidden information from large database and then generate association between the attributes in it. Market basket analysis is the common implementation in this method. Through this method it measures the dependency of each item in itemset and association rule contain with mainly two parts: an antecedent(if) and a consequent (then). An antecedent can define as item which can found within the data and consequent can define as an item which found with combine to the antecedent.

In order to identify the strength of an association rule mainly two measures 'Support' and 'Confidence' are used.

Item Sets

Collection of all the items in the given dataset. $I = \{i_1, i_2, ..., i_n\}$

Collection of all the transactions in given dataset $T = \{t_1, t_2, \ldots, t_n\}$

Every transaction is a collection of items and when there are n items it is called n itemset. If there are no items in item set it is called as null.

Support

Transaction width is defined based on the number of items in that particular transaction. Support of an item is the fraction of transaction in dataset that contain that particular item to the total number of transactions. Based on the value of support helps to consider rules which are need to consider in further analysis. Support value will allow to find the hidden relationships among the items.

Support A = Number of transactions that contain A / Total transaction (Gurudath, 2020)

Confidence

This is a measure that define the likelihood of a customer buy product X will buy product Y as well. The rule can define as a form (item set X) => (item set Y) where X is precedent and Y is Consequence. Based on the pre-existing antecedents confident provide the probability of occurrence of consequence.

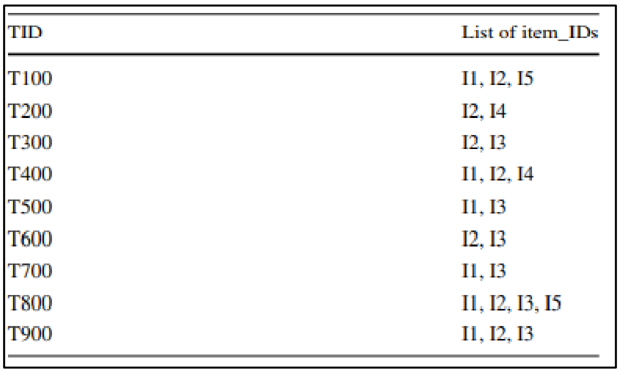
Confident (X => Y) = P(X|Y) = (Number of transaction that contain X and B) / (Total transaction that contain X) (Gurudath, 2020)

Association rules can define as a probability of relationship among the items within in a large dataset. Following are most common algorithms which use association rules based on support and confident which discussed above. These algorithms contain with different implementation procedures but have same purpose.

- Apriori Algorithm
- Frequent Pattern Growth

2.2.1.1.1 Apriori Algorithm

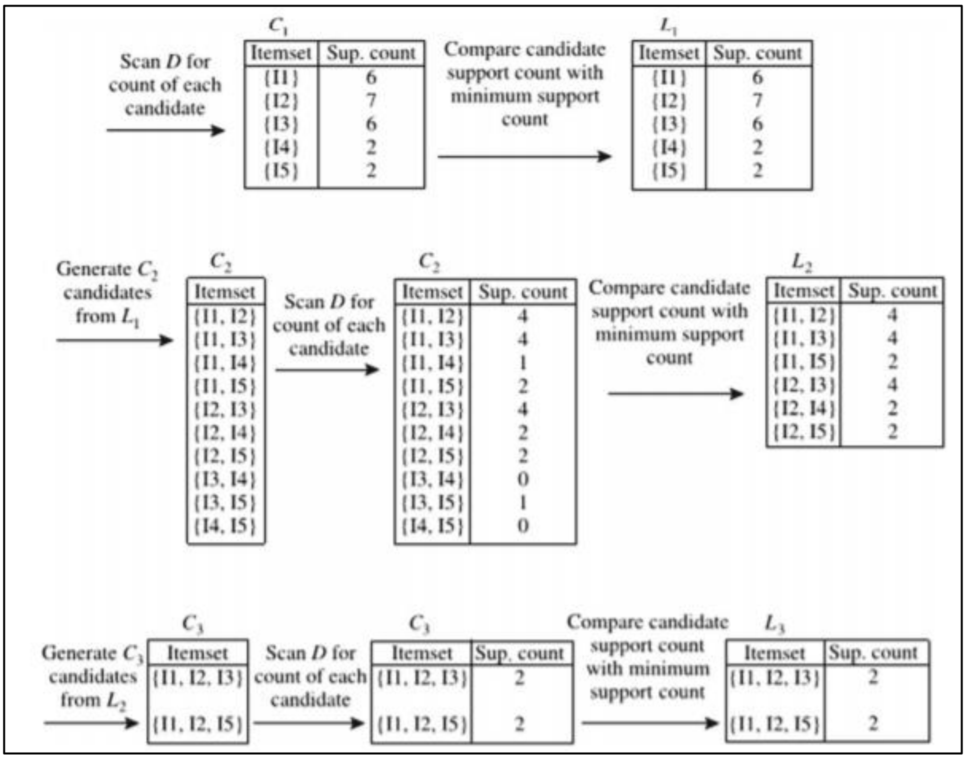
An algorithm which used for mining frequent item sets for generating Boolean association rules. Level wise search or Iterative approach was used in here where k frequent item sets are used for finding the k+1 item sets. According to the Chee et al. (2019) in following Figure 2.3 contain with sample dataset for transactions.



TID	List of item_IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Figure 2.3: Sample Transaction data (Chee, Jaafar, Aziz, Hasan, & Yeoh, 2019)

In Apriori process first database is scanned to identify all the frequent one itemset and then count each of them for capture the minimum support threshold item sets. It is required to scanning the entire database until there is no more frequent k item sets. Based on the minimum support only the records that fulfill above criteria will move into the next cycle. In following Figure 2.4 stated the result of each cycle with relate to the minimum support value.



Scan *D* for count of each candidate

C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare candidate support count with minimum support count

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generate C_2 candidates from L_1

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

Scan *D* for count of each candidate

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

Compare candidate support count with minimum support count

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

Generate C_3 candidates from L_2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

Scan *D* for count of each candidate

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare candidate support count with minimum support count

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Figure 2.4: Generation of candidate item sets and frequent item sets. (Chee, Jaafar, Aziz, Hasan, & Yeoh, 2019)

In comparing to other algorithms Apriori algorithm reduce size of candidate item sets but critical limitation such as huge candidate item sets are required to verified using different pattern matching techniques and also entire database is required to be scanned repeatedly for identifying candidate item sets.

2.2.1.1.2 Frequent Pattern Growth Algorithm

Frequent pattern growth algorithm which proposed by Han in 2000 is a scalable and efficient method that mines frequent item sets without costly process. Implement based on divide and conquer technique that convert frequent item set into frequent pattern tree. In order to mine frequent items separately the resulting FP tree further divide into set of conditional FP Trees. FP Tree is a structure which store quantitative information for frequent patterns. According to the Han's research FP tree can define as follow (Han, Pei, & Yin, 2000):

- Root labeled as 'null' with item prefix sub tree with frequent item header table.
- Frequent item header table consists with following fields such as Item-name and Head of node link

In considering the FP growth algorithm, FP Tree is generation take time to build and once it completed it will easily generate frequent item sets. After adding entire dataset to the FP Tree support can be calculated. In comparing to the other pattern mining techniques FP growth is faster than Apriori and it require only two database scans.

According to Yen et al in 2012 they combine advantages of FP growth with Apriori and generated an algorithm called Search Space Reduction. In SSR it first scans the transactions once for counting the support and identify frequent items which are higher than the minimum support threshold, then using above identified frequent item sets generate FP Tree. As a result, there is only one item-prefix tree in memory at same time. In SSR for constructing item-prefix tree use function Item-prefix-tree-construction and for candidate generation and frequent pattern generation they used function Frequent-pattern-generation.

As stated in research of Alyoubi (2020) they applied FP growth algorithm in step-by-step process where it removed unnecessary data and improve performance of overall process. Generating of the rules in FP Growth need to contain with a validation process to check whether those are applicable and have authenticity. Generated rules form FP growth can use as a recommended set of instructions which can be used in decision making process. This research use FP growth for generating hidden patterns from customer transaction data in supermarket which resulted according to (Alyoubi, 2020):

- least frequent products: can be controlled through a system
- most frequent products: products that need to be available in large quantity
- most associated products: combination of products can place together
- low confident rule: can discard these rules

2.3 Customer Segmentation

Customer segmentation has led for deeper understanding of the customer buying pattern. According to Soudagar (2012) stated that it cost five times more for gain a new customer than to keep an existing one and ten times more to get a dissatisfied customer back. By using customer segmentation as data mining can get following advantages such as (Soudagar, 2012):

- Segmented result more focus on the objectivity of the data rather than the subjectivity of the people who process them.
- Change in customer behavior can tracked based on the collocating clustering analysis models.

Clustering which is a datamining method contribute for the good exploitation and determination of the results based on analysis. As many companies more focus on improve on their marketing strategies to enhance their market share; they primarily focus on customer segmentation. By dividing the customers into clusters based on their behavior parameters can lead for a significant growth in their revenue (Ansari & Riasi, 2016). In this study it focused on 250 bank customer dataset which they found five different clusters and clusters are different based on factors such as loan amount, degree of loyalty, account balance, default risk and profitability for bank. Finding suggest that customer clustering can help for financial sector that it augments their competitiveness to improve their marketing methods to target and segment-based marketing approaches.

Smulders (2019) in his research stated that based on customer trajectories data can use to identify how customer moved inside supermarket using clustering these trajectories. They used operation edit distance which is a method to calculate similarity between two shopping paths with considering clustering on length, sequential order and spatial constraints. In this study they handled spatial constraints based on grid-based solution with A* algorithm.

2.3.1 Review of Customer segmentation based on RFM Analysis

For the customer segmentation RFM can defined as a good model based on three dimensions which are

- Recency (R) – How recently a customer has made a purchase
- Frequency (F) – How often customer makes a purchase
- Monetary (M) – How much money a customer spends on purchase

According to Aggelis and Christodoulakis (2005) in their research RFM define as a three-dimensional way that use for classifying or ranking customer which based on 80/20 principle that 20% of customer bring 80% of revenue of a company. They used RFM for study the scoring of the active e-banking users (Aggelis & Christodoulakis, 2005). In this study it used clustering as a technique for data mining and organized the finding to cluster groups based on the pyramid model as shown in Figure 2.5.

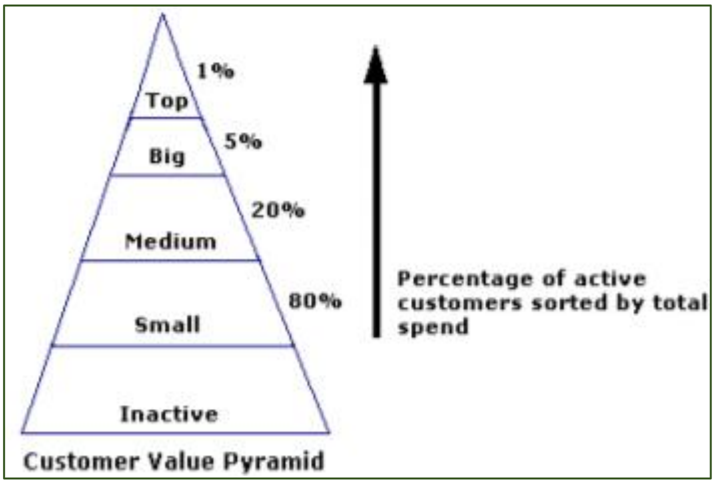


Figure 2.5: Pyramid Model (Aggelis & Christodoulakis, 2005)

In this study they used two-step clustering method and resulted bank to identify most important users-customers. By referring the study of Aggelis and Christodoulakis (2005) stated that in the above pyramid model has been useful for different businesses that it improves the issues such as

- Decision making
- Predictions for alteration of the customer position in the pyramid
- Future revenue forecast
- Customer profitability
- Simulation of inactive customers.

In past years many researches have used RFM model for prediction and development of classification models. In Table 2.1 summarize research done based on RFM analysis.

Studies	Context, research design and analysis	Purpose and key findings
Ansari & Riasi (2016)	Context: Iran 250 bank customers data RFM analysis with two step clustering	Purpose: To identify the main clusters of bank customers in order to help commercial banks to better identify their customers and design more efficient marketing strategies. Finding:

		Five different clusters namely favorite customers, creditworthy customers, non-creditworthy customers, passers and friends. Clusters are different based on their loan amount, default risk, account balance, degree of loyalty and profitability for the bank.
Chen, Kuo, Wu, & Tang (2009)	Context: Taiwan Retailing Sector RFM analysis with Apriori Algorithm	Purpose: To develop a novel algorithm based on all RFM sequential patterns from customer's purchasing data. Proposed a pattern segmentation framework to generate information for managerial decision-making.
Coussement, Van den Bossche, & De Bock (2014)	Context: Marketing Datasets provided by Direct Marketing Educational Foundation RFM analysis with decision tree and logistic regression	Purpose: To investigate the influence of problems with data accuracy for two real life data sets. Results demonstrate the impact of data accuracy on RFM analysis which recommend decision tree in context of customer segmentation for direct marketing.
Hu & Yeh (2014)	Retailing sector RFM analysis with K means clustering	Purpose: To identify RFM pattern and develop an algorithm for discover RFM patterns that can approximate set of RFM customer pattern without customer identification information. Also propose a tree RFM pattern tree to compress and store entire transactional

		database and develop RFMP-growth algorithm which based on pattern growth. Results show that approach is efficient and can use to discover the greater part of RFM customer patterns.
Jonker, Piersma, & Poel (2004)	Dutch charitable organization RFM analysis	Purpose: Proposed joint optimization approach which address following issues: Segmentation of customers into homogeneous groups of customers Determining the optimal policy Results show that model is out perform a CHAID segmentation.
Sagˇlam, Salman, Sayın, & Tu¨rkay (2006)	Satellite Broadcasting company: ‘Digiturk’ Clustering with -mean algorithm	Purpose: To present a mathematical based clustering approach with objective of minimizing the maximum cluster diameter amount the clusters. Analysis of the result indicate that the approach is computationally efficient and meaningful segmentation of data.

Table 2.1: Summary of researches based on RFM analysis

2.4 Summary

This chapter discussed the findings of the different kind of research that conducted based on the frequent itemset identification with customer segmentation and the layout design of the shopping malls. According to the finding’s retailers can allocate products effectively in their shelves. Placement of the product in the way affect to the sales of the shops and identifying the most effective way to do that is challenge that faced by the retailers. Retailers can use the travel path of each shopper and identify the frequent area which traveled most and based on that the layout can able to update accordingly.

CHAPTER 3

METHODOLOGY

This chapter highlights the solution to the above identified problem through frequent pattern growth algorithm and clustering. The study is based on the shopping transaction data, which is entered into the database and analysed using different techniques.

The following Figure 3.1 shows a general model of how the data is acquired and how the data analysis is used in order to construct the proposed solution.

- As in the figure, data collection is done in several ways, such as through the past purchase history and storing the item location-related data with other details.
- The literature review identified the algorithms which can be used for data analysis.

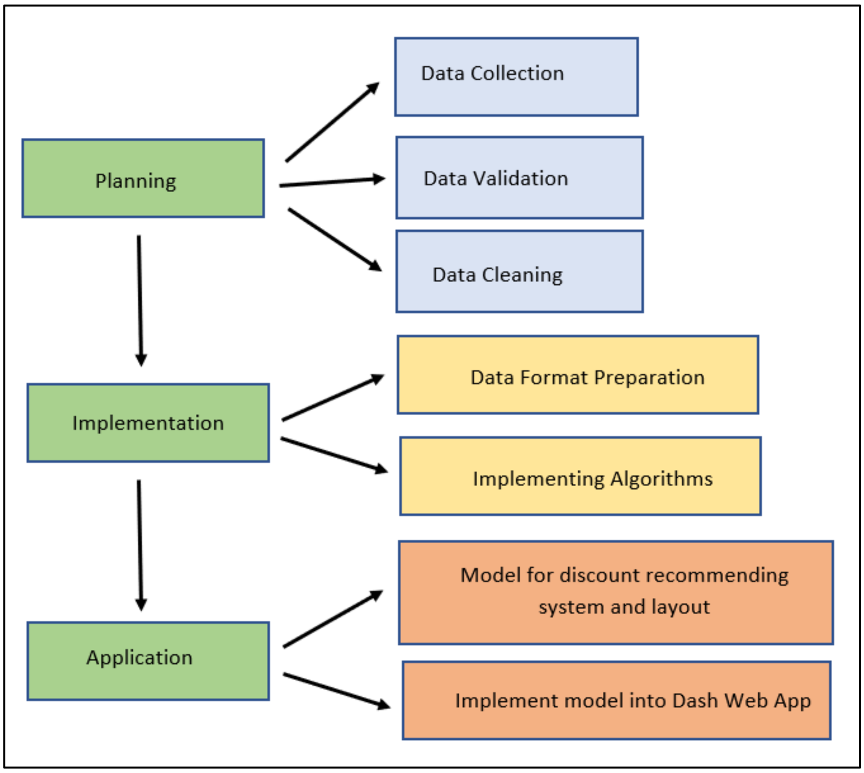


Figure 3.1: Work flow for the approach

The proposed workflow contains with stepwise manner which is stated in the Figure 3.1 that contain with main 3 phases. Each phase contains with sub phases.

3.1 Data Collection

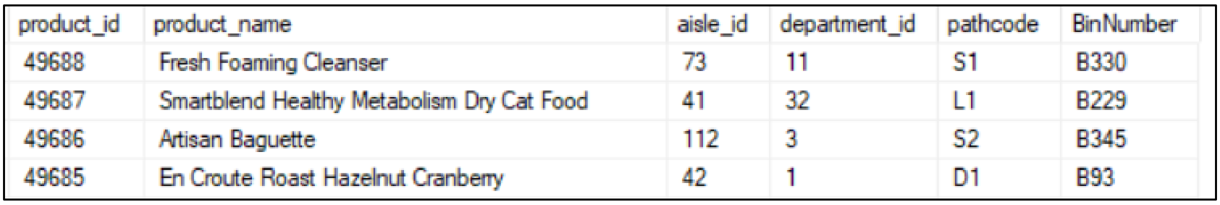
For the development of this project main dataset selected from an online datastore ‘Instacart’ which include about daily item transactions that contain with around 1 million grocery orders with around 3500 users. The details are including with transactions based on the customer, products which they bought, departments that the products are associate with and aisles with the bin where the products are placed.

Data	*Contain*
Products	Items in the store
Categories	Categories which items are belong
Aisles	Rack and Bin ID of Products
BinLocation	Bin location codes which each item is assigned
Orders	Transactions placed by customers

Table 3.1: Detail about tables

Products

Table contain with details about products. Table include columns such as product_id, product_name, aisle_id, department_id, pathcode and BinNumber. Product_id is unique for a product.

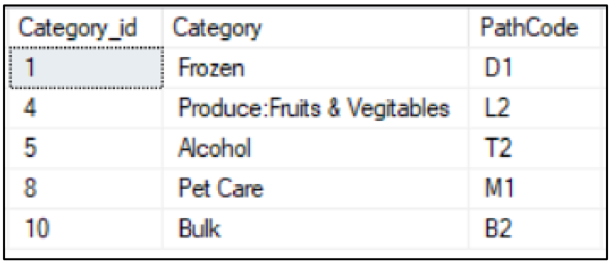


product_id	product_name	aisle_id	department_id	pathcode	BinNumber
49688	Fresh Foaming Cleanser	73	11	S1	B330
49687	Smartblend Healthy Metabolism Dry Cat Food	41	32	L1	B229
49686	Artisan Baguette	112	3	S2	B345
49685	En Croute Roast Hazelnut Cranberry	42	1	D1	B93

Figure 3.2: Products Table

Categories

This table contain mainly with 21 categories contain with 55 subunits. Table include details such as category_id, category and path code. Path is the location which the aisle of category is located.



Category_id	Category	PathCode
1	Frozen	D1
4	Produce:Fruits & Vegitables	L2
5	Alcohol	T2
8	Pet Care	M1
10	Bulk	B2

Figure 3.3: Category Table

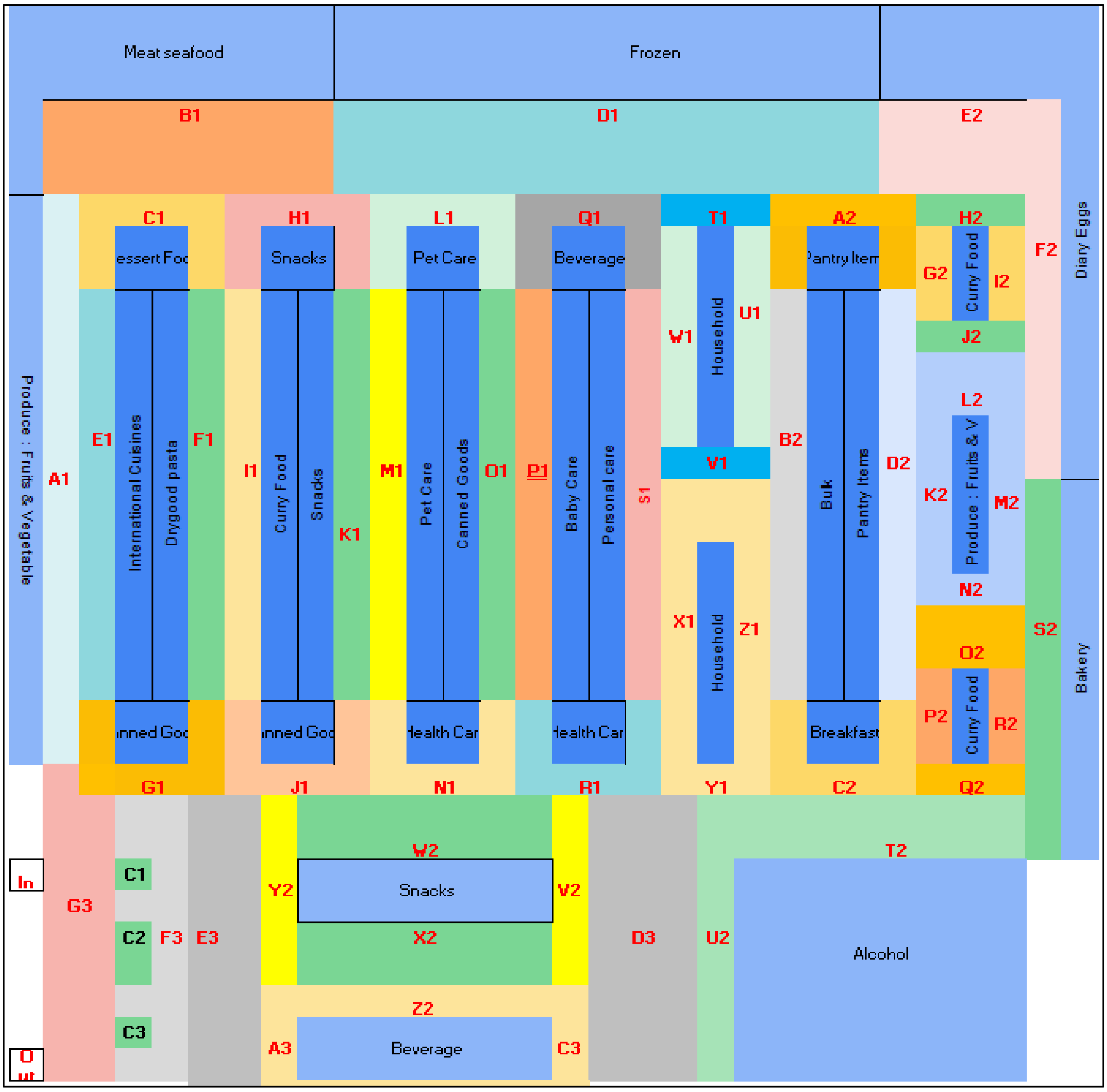
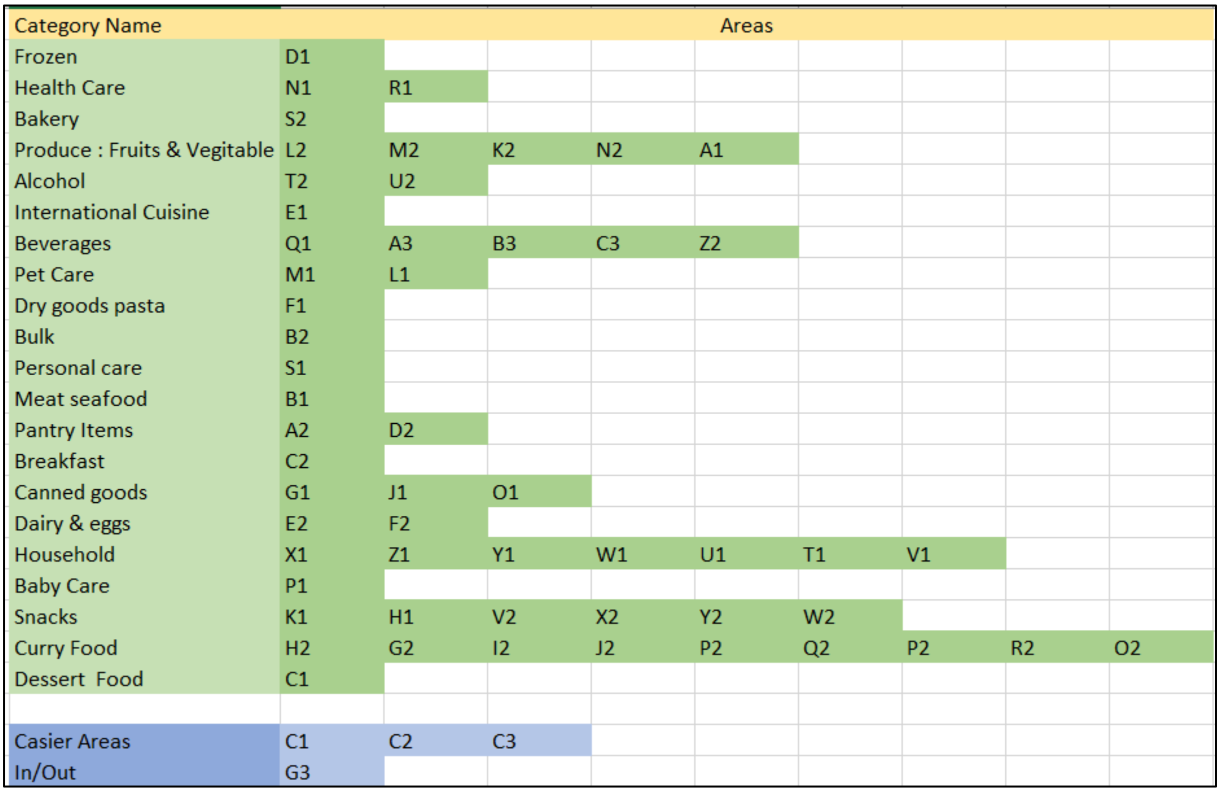


Figure 3.4: Layout of the shopping mall

Above Figure 3.4 illustrate the layout of the shopping mall with each area assigned with a particular code. This code was used to track the path of the user in traveling inside the shop.



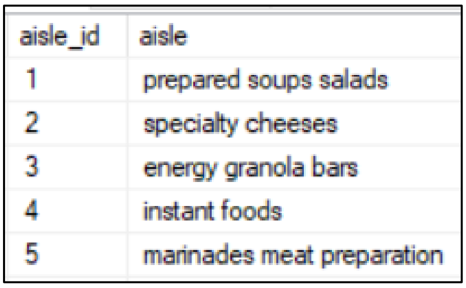
Category Name	Areas								
Frozen	D1								
Health Care	N1	R1							
Bakery	S2								
Produce : Fruits & Vegitable	L2	M2	K2	N2	A1				
Alcohol	T2	U2							
International Cuisine	E1								
Beverages	Q1	A3	B3	C3	Z2				
Pet Care	M1	L1							
Dry goods pasta	F1								
Bulk	B2								
Personal care	S1								
Meat seafood	B1								
Pantry Items	A2	D2							
Breakfast	C2								
Canned goods	G1	J1	O1						
Dairy & eggs	E2	F2							
Household	X1	Z1	Y1	W1	U1	T1	V1		
Baby Care	P1								
Snacks	K1	H1	V2	X2	Y2	W2			
Curry Food	H2	G2	I2	J2	P2	Q2	P2	R2	O2
Dessert Food	C1								
Casier Areas	C1	C2	C3						
In/Out	G3								

Figure 3.5 Area code/s for each category in the shopping mall.

Figure 3.5 illustrate the area code/s for each category and these codes are unique which contain 55 unique values.

Aisles

This table contain with 134 aisles with including details such as aisle_id and aisle. Following are sample details of aisle records. Each aisle is assigned with different products based on products category.

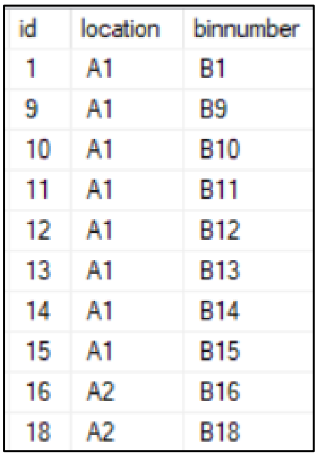


aisle_id	aisle
1	prepared soups salads
2	specialty cheeses
3	energy granola bars
4	instant foods
5	marinades meat preparation

Figure 3.6: Aisle table

Bin Location

Table contain with details about Bin locations which the items are assigned in each aisle. Each aisle is divided into several bins and each bin is assigned with a code. Table include columns such as id, location and bin_number. Figure 3.7 visualized sample of bin location codes.

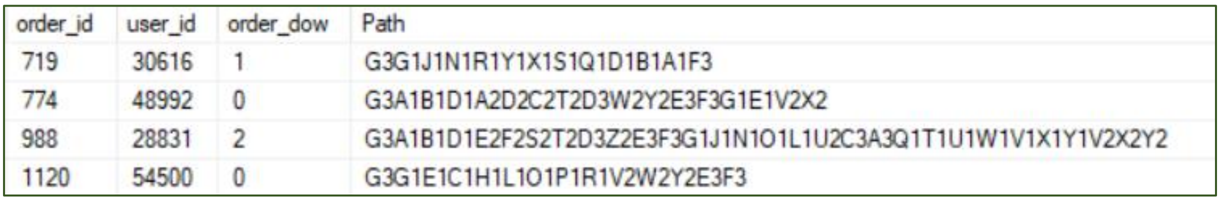


id	location	binnumber
1	A1	B1
9	A1	B9
10	A1	B10
11	A1	B11
12	A1	B12
13	A1	B13
14	A1	B14
15	A1	B15
16	A2	B16
18	A2	B18

Figure 3.7: BinLocation Table

Orders

Table contain with details about orders which placed by the users. For a one user there are range of 4 to 100 of orders. Table include columns such as order_id, user_id, order_dow and path.



order_id	user_id	order_dow	Path
719	30616	1	G3G1J1N1R1Y1X1S1Q1D1B1A1F3
774	48992	0	G3A1B1D1A2D2C2T2D3W2Y2E3F3G1E1V2X2
988	28831	2	G3A1B1D1E2F2S2T2D3Z2E3F3G1J1N1O1L1U2C3A3Q1T1U1W1V1X1Y1V2X2Y2
1120	54500	0	G3G1E1C1H1L1O1P1R1V2W2Y2E3F3

Figure 3.8: Orders Table

Orders table is containing with the ‘path’ column which store the data related to the path that the user is traveled inside the supermarket for that particular order. These path codes are to track the location assigned for a particular item category. Based on that path is generated.

3.2 Data Validation, Data Cleaning and Data Format Preparation

The original dataset was obtained from the Kaggle website and it contains around 3 million records with 200,000 users. For this project, selected only around 3000 users and their orders because of the hardware limitations. As the data was stored in comma separated value (csv) files, those are imported into the Structured Query Language (SQL) server. Based on the data in each column, the data format is updated accordingly. Before moving into the analysis of the data, it is pre-processed.

Columns which affect the results selected for the further process. The dataset contained missing or incomplete values. In order to replace the missing values in the original dataset, a method called imputation has been used. Imputation is a method that replaces the missing value with non-null values. To identify the missing values, a method called 'isnull()' is used.

3.3 Implementing Algorithm

3.3.1 Frequent Pattern Mining Algorithm

Han in 2000 introduced the method FP growth, which is a scalable and efficient method that can be used for mining frequent patterns using extended prefix-tree structure. In an FP tree, every branch represents a frequent itemset and each node along the branches is stored in decreasing order of the corresponding item's frequency. In a branch, leaves represent the least frequent items. FP tree is defined as a compressed representation of the itemset of the database.

FP Tree representation:

FP tree is known as a collection of tree shaped records with a compact data structure. Each transaction in the database is read and sorted into an FP tree path, which will continue until all transactions are read out. Generally, this algorithm is designed to operate on datasets that contain transaction data such as purchase orders by customers. An item is considered as 'frequent' if it satisfies the user specified support threshold value.

As an example, if support is defined as 0.7 (70%), a frequent itemset is defined as items which occur together in at least 70% of all the transactions.

	FP Growth	Apriori
Speed	Runtime increase in align to the number of item sets	Runtime increase exponentially align to the number of item sets.
Frequent Pattern	Based on mining conditional FP tree pattern growth achieved	Select patterns which are higher than the minimum support defined.
Scan	Two database scans	Throughout the process database is scanned.
Memory	Store a compact version of database	All the self-joined candidates are stored.

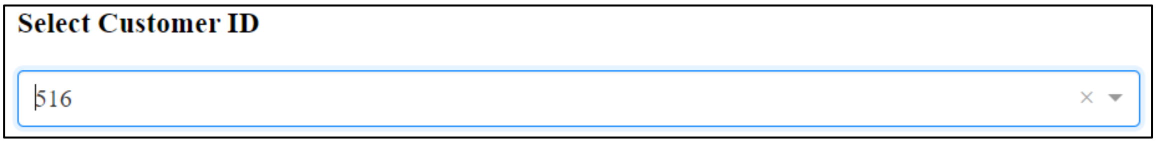
Table 3.2: Compare FP Growth vs Apriori

In this research FP growth algorithm is apply to identify the most frequent path of a particular customer and to identify the most frequent path considering the all customers.

3.3.1.1 Applying FP Growth Algorithm

Step 01: Identify the most frequent path for a particular customer

In web application dropdown list allow to select a particular customer and based on that transaction data for that customer is selected (Figure 3.9).

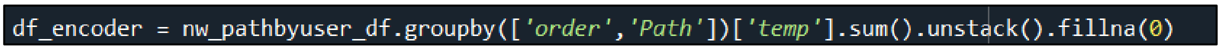


Select Customer ID

516

Figure 3.9: Dropdown for Customer code selection

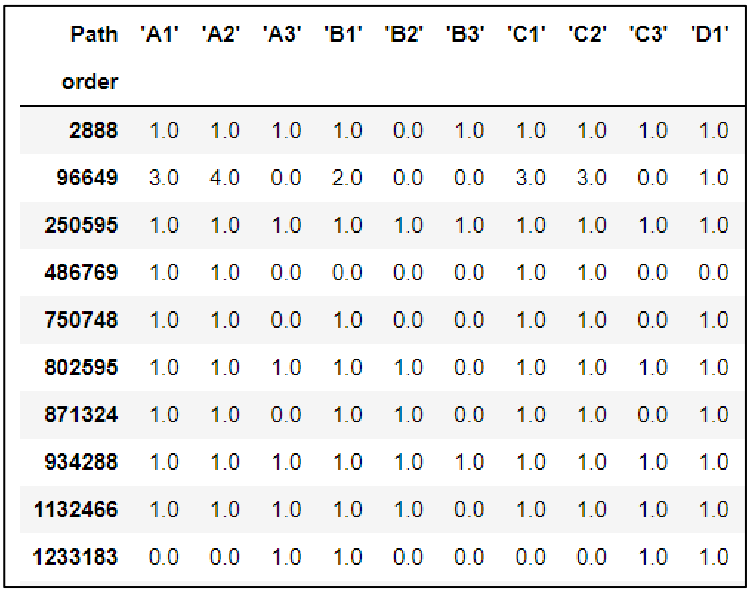
According to the Figure 3.10 selected transaction dataframe for the customer ‘516’ is used for further analysis.



```
df_encoder = nw_pathbyuser_df.groupby(['order','Path'])['temp'].sum().unstack().fillna(0)
```

Figure: 3.10: Code for switching row to column of dataframe

Figure 3.10 illustrate the converting of the filtered dataset into switching row to columns in order to get the sum as features. Using of the unstack allow to transforming the index into columns. As according to the Figure 3.10 ‘fillna()’ method used to fill the cells with missing values. As an example, for a particular order if a location code is not visited that cell will fill with zero while visited cells filled with ‘1’ (In Figure 3.11 consider order ‘750748’ Path ‘A3’).



Path	'A1'	'A2'	'A3'	'B1'	'B2'	'B3'	'C1'	'C2'	'C3'	'D1'
order										
2888	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0
96649	3.0	4.0	0.0	2.0	0.0	0.0	3.0	3.0	0.0	1.0
250595	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
486769	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0
750748	1.0	1.0	0.0	1.0	0.0	0.0	1.0	1.0	0.0	1.0
802595	1.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0
871324	1.0	1.0	0.0	1.0	1.0	0.0	1.0	1.0	0.0	1.0
934288	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1132466	1.0	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0
1233183	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0

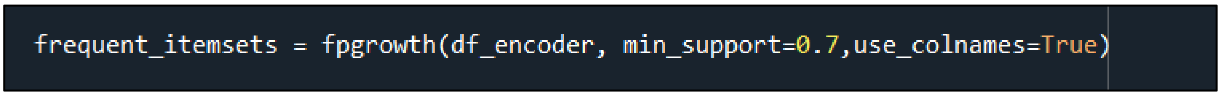
Figure 3.11: Resulting dataframe after code in figure 3.10

Above resulting dataframe is converted as follow:

If cell value is equal to zero -> replace cell value with zero

If cell value is higher than zero -> replace cell value with one

Figure 3.11 dataframe is converted as according to above criteria and that dataframe is used for frequent path generation according to Figure 3.12.



```
frequent_itemsets = fpgrowth(df_encoder, min_support=0.7,use_colnames=True)
```

Figure 3.12: Code for FP growth algorithm apply

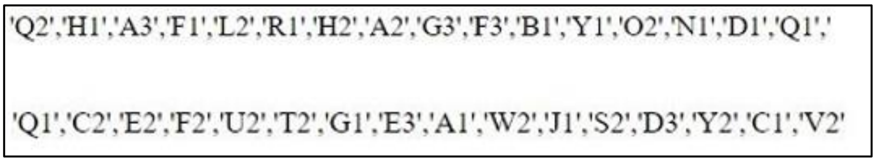
Above method result the frequent item sets based on the following parameters.

df_encoder : dataframe which is in encoded format.

min_support: support value which is between 0 to 1 in order to filter the item sets. Support is calculated based on the fraction of transaction where the item(s) occur/ total transactions.

use_colnames: Allow to use passing dataframe column names into the resulting dataframe.

Based on the defined parameters, the frequent path for the selected customer ‘516’ is as followed in Figure 3.13. Frequent paths are separated using commas in order to identify the location clearly. In this result, it visualized only the frequent travel path, but customers may not have purchased items in some of the locations.



```
'Q2','H1','A3','F1','L2','R1','H2','A2','G3','F3','B1','Y1','O2','N1','D1','Q1','

'Q1','C2','E2','F2','U2','T2','G1','E3','A1','W2','J1','S2','D3','Y2','C1','V2'
```

Figure 3.13: Frequent path codes for Customer ‘516’

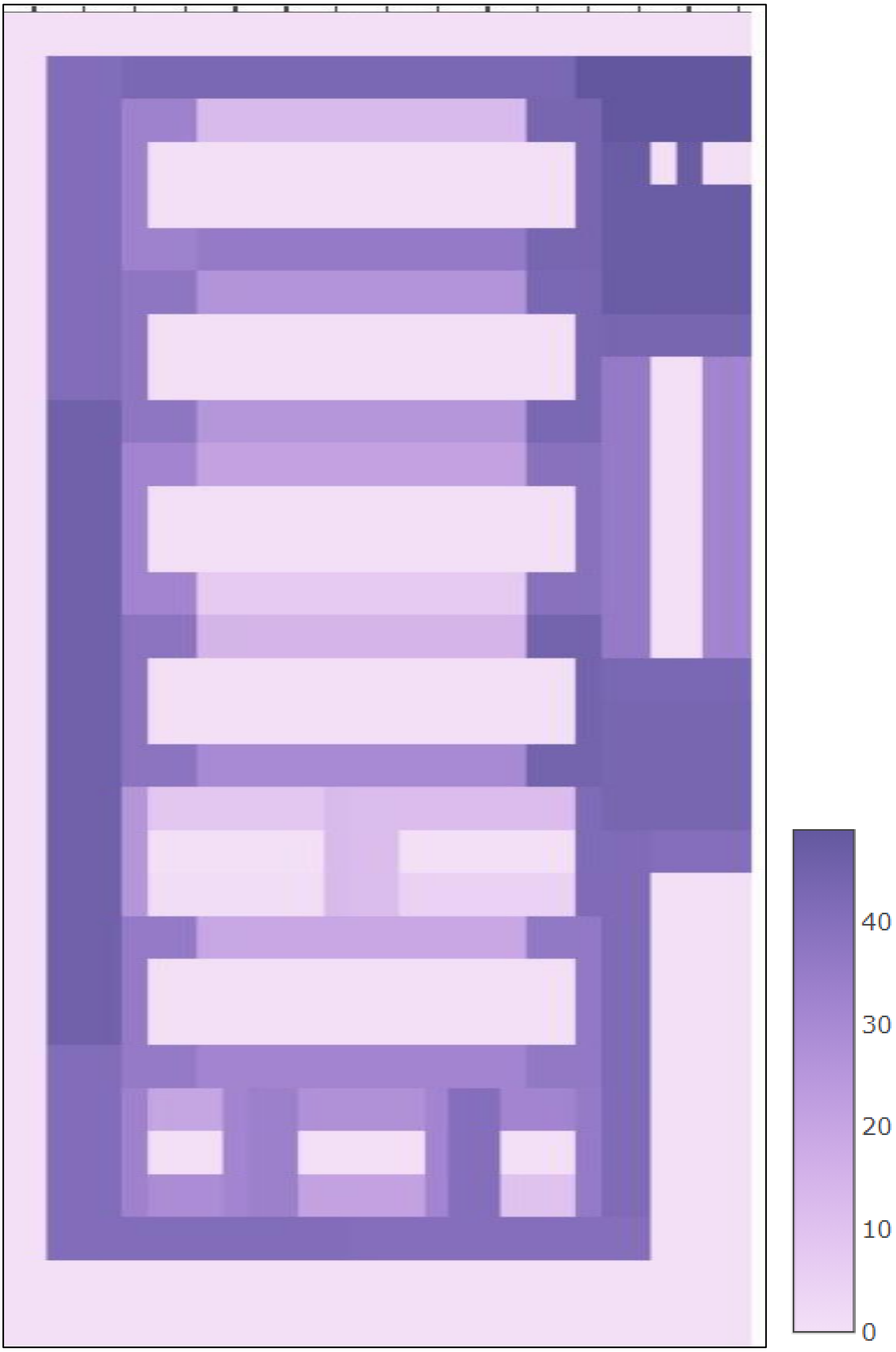
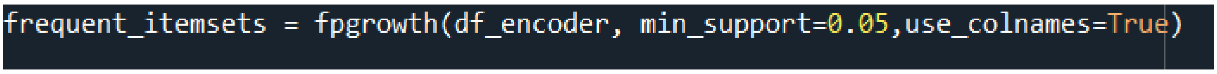


Figure 3.14: Heatmap for traveled path of Customer '516'

Figure 3.14 illustrates the heatmap for the frequent travel path of customer '516'. In the heatmap dark color areas are the most visited areas and light color areas are the least visited by the customer (white color areas are aisle locations).

Step 02: Identify the most frequent items in the most frequent path of the customer '516'

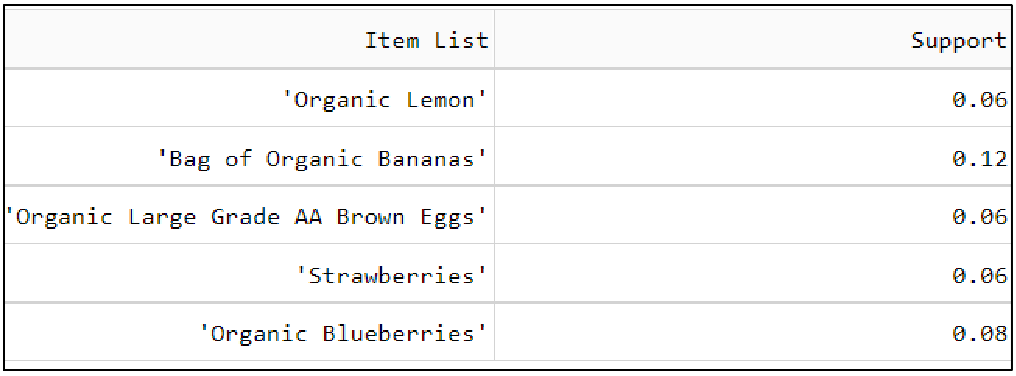
Based on the above identified frequent path, frequent item set is generated. For identification of the frequent item list FP growth algorithm was used. Products which are located in the frequent path was selected from the database and that dataframe is used for further analysis as the steps discussed on the step 01.



```
frequent_itemsets = fpgrowth(df_encoder, min_support=0.05,use_colnames=True)
```

Figure 3.15: FP growth for frequent item identification

Figure 3.15 illustrate the applying of FP growth for the frequent itemset generation based on the frequent path identified previously.



Item List	Support
'Organic Lemon'	0.06
'Bag of Organic Bananas'	0.12
'Organic Large Grade AA Brown Eggs'	0.06
'Strawberries'	0.06
'Organic Blueberries'	0.08

Figure 3.16: Frequent item list of customer '516'

Figure 3.16 illustrate the frequent item list for the frequent path in Figure 3.13 for the customer '516' which identified using the FP growth algorithm. Support value for each item is calculated using FP growth. According to the result can identify product 'Bag of Organic Banana' is the most frequent purchased item of customer '516'.

3.3.2 Clustering

Clustering is the process of grouping objects based on their similarity. Objects that are in the same group are similar to each other more than the objects in the other groups. Algorithms such as K-Means and K-Medoid are famous techniques that come under the category of clustering techniques. As the real data is bimodal, that means a joint interaction between two variables. Co-clustering is the simultaneous clustering of rows and columns in a matrix based on their similarity to other objects of the same type. If there are ‘n' objects in the original data set, the original data will be divided into k partitions.

- Based on the closest centroid, assign each object to a cluster.
- Based on the mean value of the objects, compute the new position of each centroid.
- The meanings are fixed above and do not need to be repeated.

As the dataset contains a large number of orders, technical limitations lead to clustering the users according to similarities. Based on these clusters, a special discount was suggested and it will be discussed under section 3.4. Through this research, RFM analysis was used in order to cluster the customers based on their similarities before a certain discount could be assigned.

RFM Analysis

RFM analysis is a strategy that analyses the customer based on three factors: Recency, Frequency and Monitory values. It is a proven strategy for customer segmentation. In applying RFM to a dataset, it is needed to identify relevant fields for R, F and M. In this study, RFM analysis is used to cluster the customers in order to identify their behaviours as a group. Based on the characteristics of the three factors, the following fields are identified from the database to identify the customer segments.

Recency: Total number of orders per customer

Frequency: Average days between the orders per customer

Monitory: Average size of orders per customer

Total orders and average days between orders per customer are similar to the recency and frequency in the RFM. These features capture how much a particular user is using Instacart. For the monitory value, the average size of the orders per customer is considered because it can get through the quantity.

Based on these three measures, it will be possible to identify how many clusters are most suitable to divide the given dataset into. Identification of the best k value for the k-mean algorithm is important as it leads to minimizing the effect of outliers.

Identification of the best number of clusters(k) can evaluate based on several metrics such as:

Distortion score:

Measure the distance between centroid to the each datapoint and lower distance score define a tighter cluster and common features.

Silhouette score:

Compare the distance of any datapoint to the center of its assigned cluster and the distance of that datapoint to the center of other clusters. Lower value define that the cluster is tighter and farther from other clusters.

Applying the clustering to this selected dataset can identify the behavior of the customers. Based on the behavior patterns, customers can be divided into subgroups or segments. These subsegments provide an insight into the details hidden inside these customers' buying and travel patterns. Analyzing subgroups will help to identify the behavior that is specific to them rather than analyzing the entire group at once. If analyze the full dataset at once, important findings will be lost. Also, considering the limitations of the computational power, it is better to analyze the segment at one time.

Using the three measures of RFM analysis in clustering support, to identify how each subgroup is differentiated based on the number of orders done, how often these customers shop, and how their order size is differentiated per customer. Based on these features, clustering will allow us to segment the customers, and these findings will help us develop a customer specific discount for each segment.

Step 03: Identify the Clusters of Customers

Based on the features above defined in the RFM analysis, a suitable dataset was selected. The selected dataset was inserted into the k-mean algorithm and model performance was calculated based on the silhouette score. The resultant silhouette score for the number of clusters two to nine was visualized in the following Figure 3.17. A significant score change happened after cluster number four, according to the graph. The best k value for the cluster is four based on the definition of silhouette score, but for further clarification, the distortion score is also considered.

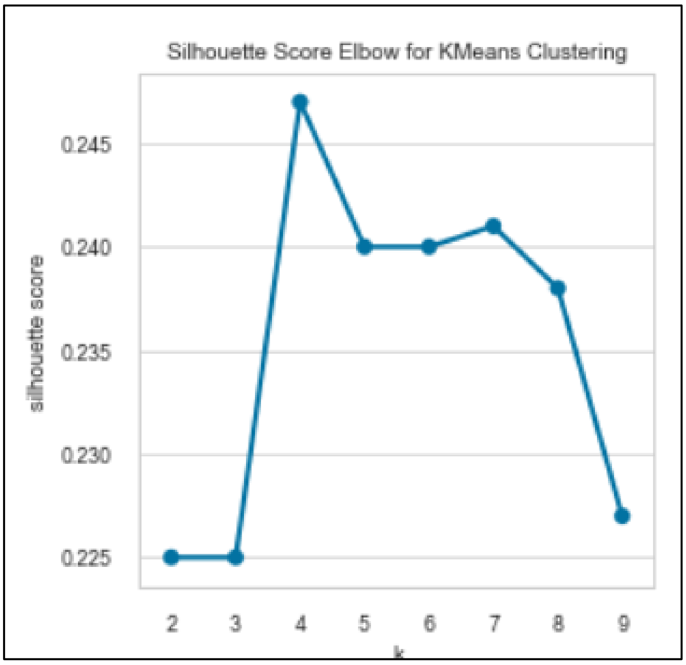


Figure 3.17: Silhouette Score Graph

Distortion score was also used in order to clarify the best number of clusters of customers which were identified under the silhouette score. In this method, the K-elbow visualizer is implemented based on the ‘elbow’ method in k-mean clustering. As an unsupervised machine learning algorithm, ‘k-mean' groups the data into k clusters. In this scoring method, the user must specify in advance the range of clusters, and the elbow method computes the average score for each cluster. According to the score plotted in Figure 3.18, elbow point k=4 can be defined as the best k value. If there is a strong inflection point, it is defined as a good indication of which model fits best at that point.

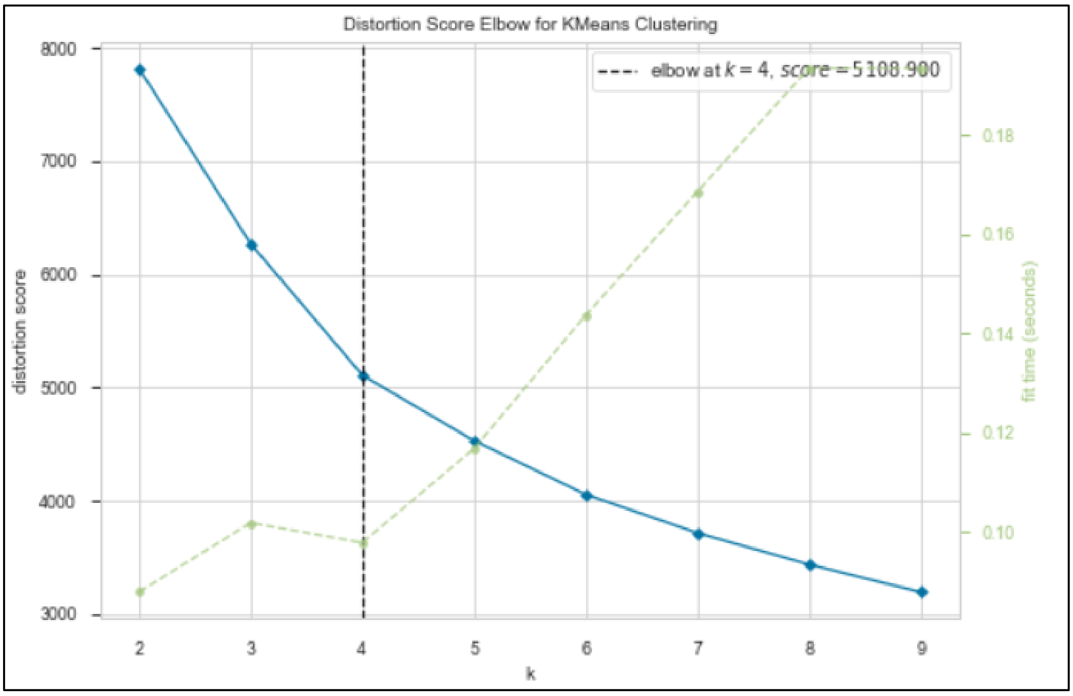


Figure 3.18: Distortion Score Graph

According to the Lim (2019) stated that it is a common practice to proceed with not with only the best ‘k’ value but also with ‘k-1’ and ‘k+1’ also. Based on the Figure 3.17 and Figure 3.18 best k value for the clustering user was identified as the four.

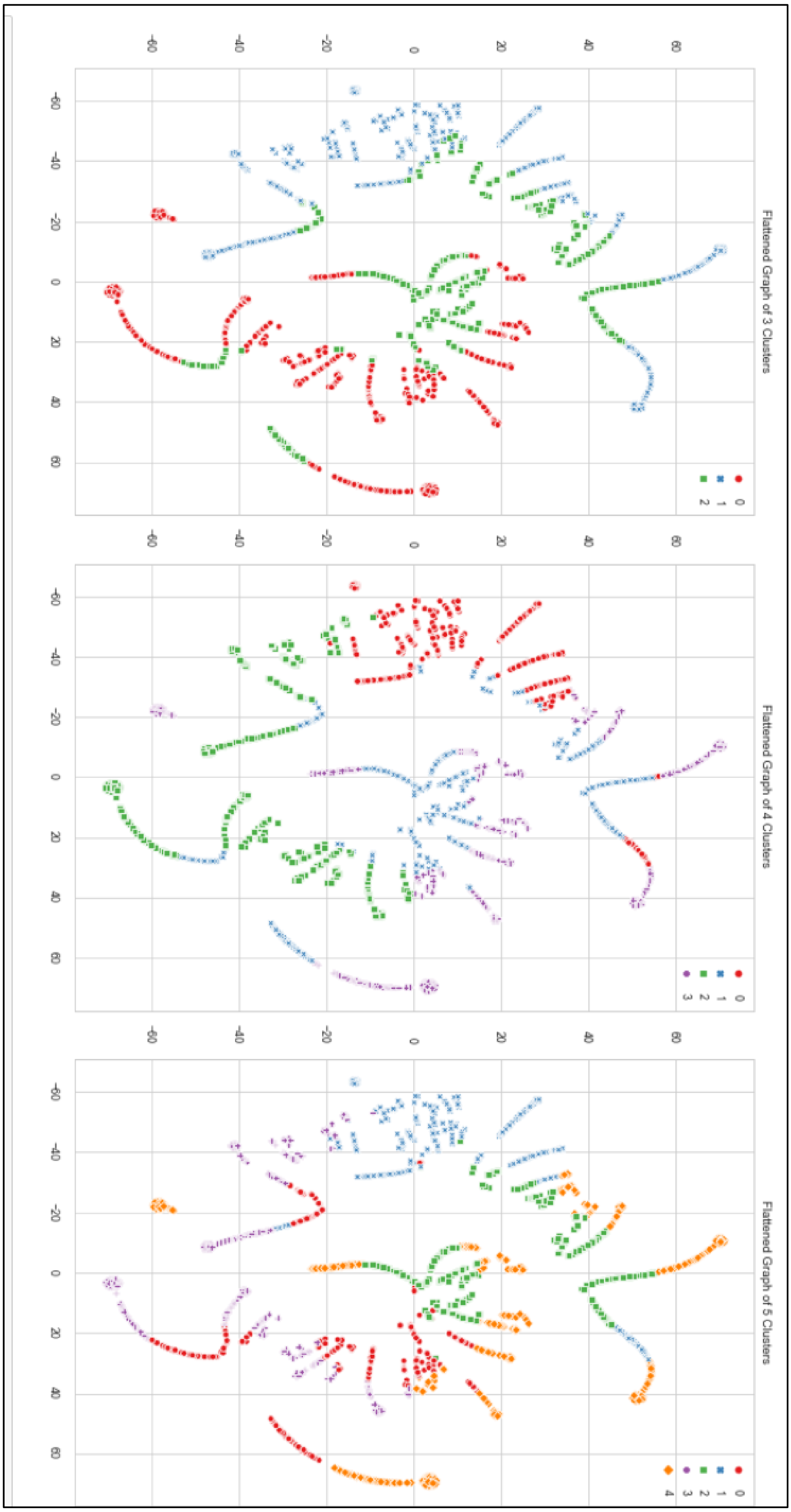


Figure 3.19: Flattened Graphs based on k=3, k=4, k=5

Figure 3.19 illustrate the customer clustering based on k=3, k=4, k=5 and based on each graph k= 4 has a clear separation of points in to separate clusters.

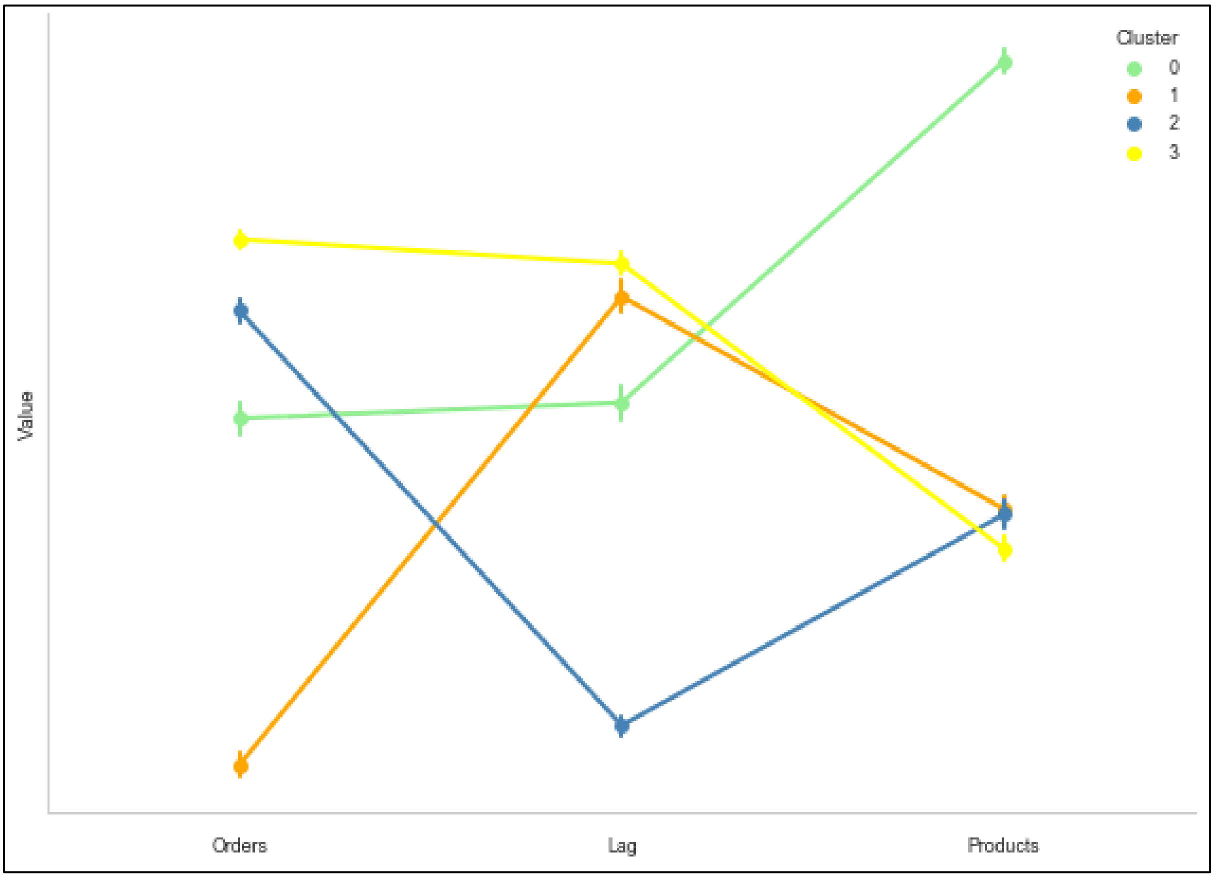


Figure 3.20: Snake Plot Graph for 4 clusters

Based on the finding of the above methods k=4 identified as the best k value for customer segmentation. Figure 3.20 illustrate snake plot graph to visualize the average value of main three features which identified in R, F and M for each cluster.

Cluster 0: Customers who place average amount of orders with having average visiting rate but large number of products in orders.

Cluster 1: These are the customers who place lowest order rate but not visit the shop often and once visit placed average number of products in order.

Cluster 2: This segment of customers places more orders but visit the shop often and as they often visit the shop each order it contain average number of products.

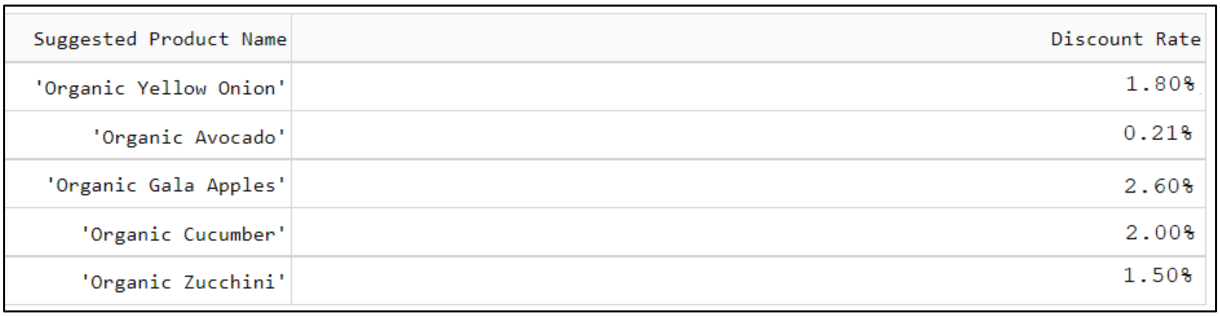
Cluster 3: Customers in this segment can identified as most order placing segment but not frequent with least number of products in the orders when comparing to the other segments.

In comparing these segments, cluster 2 customers can identify as most visiting customer which bring more revenue to the shop comparing to the other customer segments. These customers can identify as the segment who travel through the shopping layout more often and focusing on this group in new layout development will help to.

Step 04: Identify the discount applicable item for customer '516'

As stated in the previous step, every customer in the dataset was assigned to a particular cluster based on three factors, such as the number of orders, average days between orders, and the average size of the orders per customer. Items which are suggested with discounts are the items which are purchased by other customers but may or may not by this particular customer and which are located on his/her most frequent path.

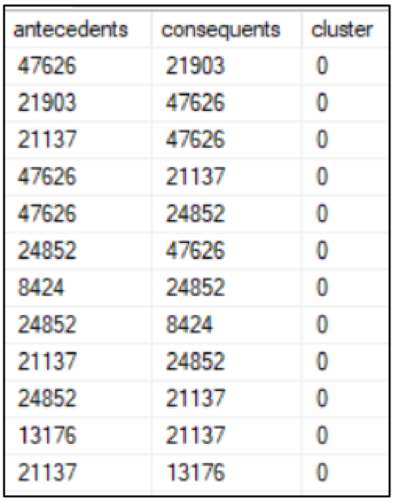
For the customer '516' discount, the applicable item list is illustrated in Figure 3.21. The calculation mechanism of the discount rate is discussed under section 3.4.



Suggested Product Name	Discount Rate
'Organic Yellow Onion'	1.80%
'Organic Avocado'	0.21%
'Organic Gala Apples'	2.60%
'Organic Cucumber'	2.00%
'Organic Zucchini'	1.50%

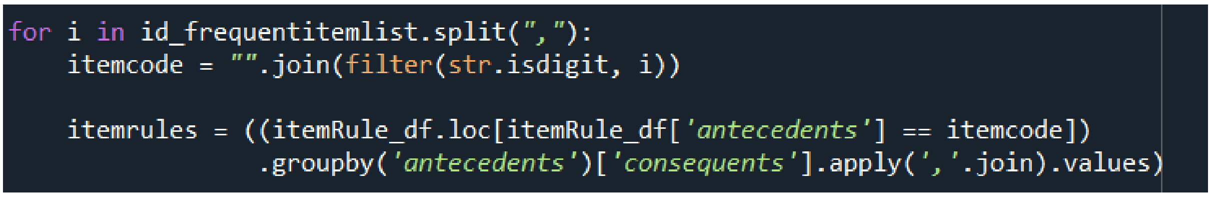
Figure 3.21: Discount applicable item list for customer '516'

In step 01 identified the frequent path of the customer using FP growth algorithm. Using FP growth algorithm item rules can generated on time but as because of the computational limitations item rules with indicate about the antecedents and consequents for each cluster are calculated and stored in SQL server as state in Figure 3.22.



antecedents	consequents	cluster
47626	21903	0
21903	47626	0
21137	47626	0
47626	21137	0
47626	24852	0
24852	47626	0
8424	24852	0
24852	8424	0
21137	24852	0
24852	21137	0
13176	21137	0
21137	13176	0

Figure 3.22: Item rules table



```
for i in id_frequentitemlist.split(","):
    itemcode = "".join(filter(str.isdigit, i))

    itemrules = ((itemRule_df.loc[itemRule_df['antecedents'] == itemcode])
                 .groupby('antecedents')['consequents'].apply(','.join).values)
```

Figure 3.23: Filter item rules based on frequent items

Based on the code in Figure 3.23, item rules are filtered for frequent items. Further filtering is carried out in order to select items in customer's frequent path. These filtered item rules are displayed as the discounted items that a customer is likely to purchase during his next visit to the supermarket. ‘Antecedents’ are the items that a customer has already bought, and ‘Consequents’ are the items that a customer will likely to buy if he/she bought the items in ‘Antecedents’. Figure 3.21 result is generated based on these steps.

3.4 Recommendation of discount and layout

3.4.1 Formular for discount recommendation

Increasing of the sales is primary objective of the business. For this purpose, most of the shops use different way and discounting method is one of them. Introducing a discount value for products based on the travel path of user is a one objective of this project. For discount calculation following formulas are used.

$$Discount\ Rate = \left(\frac{Expected\ Rate}{Current\ Rate}\right)^{1/t} - 1$$

t – Number of Years (Thakur, n.d.)

$$Average\ Purchase\ per\ customer\ for\ Product\ 'ItemName' = \frac{Total\ Quantity\ for\ the\ product}{Number\ of\ customers\ bought}$$

Based on following scenarios discount is calculated accordingly. (Consider the customer ‘516’):

- Based on above calculation if a customer’s purchase quantity is less than the ‘average purchase per customer’ then calculate discount to move him/her to average level.

  Average purchase per customer for product ‘Organic Avocado’ = 7490 / 3049

  = 2.42

  Average purchase for ‘516’ for product ‘Organic Avocado’ = 2

  By considering above calculation for customer ‘516’ his/her discount for that item can calculated based on 1st equation (if t =1 year):

  $$= (2.42 / 2)^{1/1} - 1 = 0.21\%$$

  Discount for the item ‘Organic Avocado’ is 0.21%

- If customers purchase rate is higher the ‘Average purchase per customer’ or equal, then fixed rate of discount is assigned based on customer segmentation which identified under clustering.

  Average purchase per customer for product ‘Organic Yellow Onion’ = 4290 / 2449

  = 1.75

  Average purchase of customer ‘516’ for product ‘Organic Yellow Onion’ = 4

  By considering above calculation can identify that for customer ‘516’ his/her discount for that item is a fixed rate as because their purchase rate is above average.

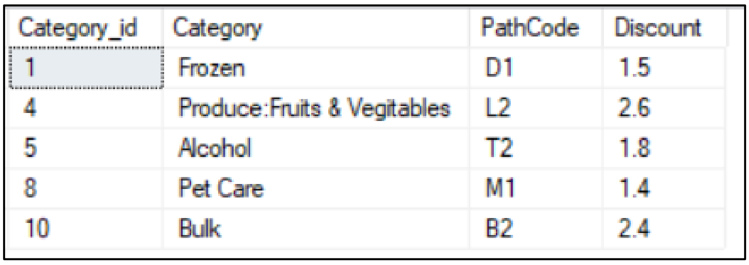
- Some items in the discount list may be not bought by the customer previously. For those kinds of items, discount is a fixed rate.

Product ‘Organic Gala Apples’ is never bought by customer ‘516’.

Average purchase per customer for product ‘Organic Gala Apples’ = 2809 / 1932

= 1.45

These fixed discounts rate for each category is defined in the database as accordingly as stated in Figure 3.24.



Category_id	Category	PathCode	Discount
1	Frozen	D1	1.5
4	Produce:Fruits & Vegitables	L2	2.6
5	Alcohol	T2	1.8
8	Pet Care	M1	1.4
10	Bulk	B2	2.4

Figure 3.24: Category table with Discount

3.4.2 Model for layout recommendation

According to industry estimates, a typical store requires remodelling of its layout every ten years after it opens and every six to seven years to improve its looks, efficiency, and operations. This layout remodelling needs to be based on the regular customers' purchasing behaviour, as they are the segment that brings more revenue to the shop rather than infrequent customers.

Under step 03 in section 3.3.2, we identified four customer segments based on RFM analysis. These segments are identified based on three measures which are:
Recency: The total number of orders placed by a single customer.
Frequency: Average days between the orders by a single customer
Monitory: Average size of orders per customer.

Based on these three factors, RFM analysis resulted in four clusters (in referring to Figure 3.19) and among those clusters, cluster two can be identified as the most important segment for the shop. Because that segment includes the high order placing rate, the minimum days between orders, and the average size of orders. Layout updating needs to consider this segment as they are the regular visitors and they may know most of the areas in the layout because they have an average size of orders.

Because of these reasons, customer segment two is used for the layout recommendation in this project. Through the frequent path, which is identified in section 3.3.1.1, demonstrates a method to find the most travelled areas of a customer. This only includes most traveling areas, but not all traveling areas of the customer. If the least travelled areas include the most travelled areas, customers may tend to buy new products as they travel through the shop. This may result in improved sales and the expansion of the traveling areas of the customer. For the layout update following steps were used in recommendation process.

Step 01:

In Customer segment 02 top 100 customers are used for this process as it represents the behavior of that full cluster and also to overcome the technical limitations which will face of executing full dataset at once.

Step 02:

By applying FP growth for these customers identified the most traveling areas of these customers.

Step 03:

Based on the above result; items which are in the most frequent traveling path and bought by customers have identified (I_1).

Step 04:

From above result identified the infrequent locations which customers are least traveled.

Step 05:

Aisles relate to these infrequent locations are relocated to the frequent path of the customers.

Step 06:

Identified the items that are in the relocated aisles which are bought by the customers (I_2).

Step 07:

Based on the step 03 and step 06 summarized that there are new items in the frequent path which are already bought by customers(I_3).

$I_3 = I_1 - I_2$

When items are in the frequent path of the customers, they will tend to buy products than before. By following those steps, can identified that changing of the locations of some of the aisles will result to improve the frequent path and also to move infrequent items to frequent items.

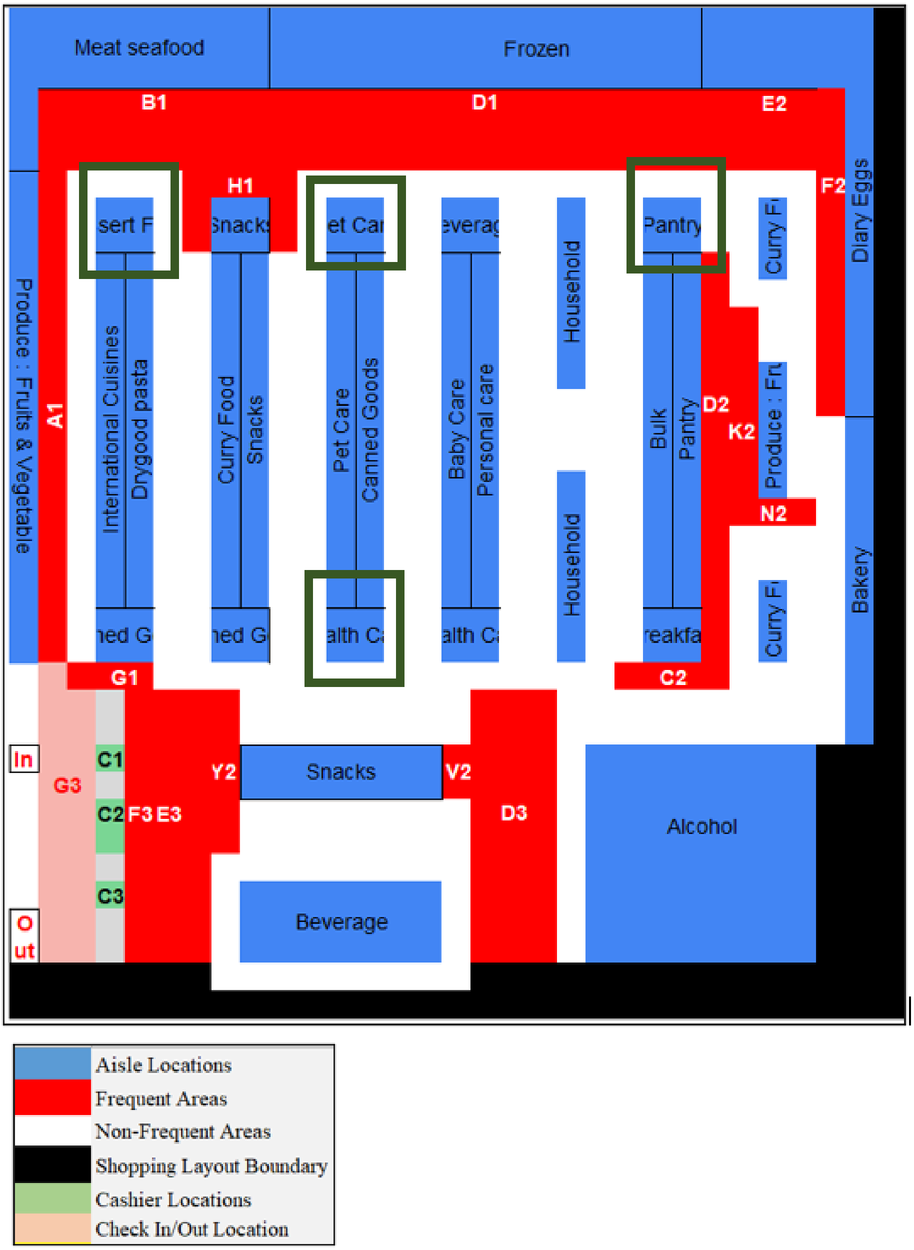


Figure 3.25: Location to be update in current layout

In Figure 3.25, areas selected through rectangles are the locations that have fewer frequent visits. Changing these locations based on the steps that have been defined previously will help to move the non-frequent sale items into frequent travel areas. Through that, it will help customers select those items easily while those are in frequent travel paths and will improve the travel frequency for those areas.

3.5 Implementation: Prerequisites

Implementation of the project was done using python as programming language and SQL server for data store. In parallelly following software and libraries are also used.

Software

- Windows OS 64 bit
- Anaconda 64 bit
- Python 3.7
- Spyder 4.1
- Jupiter

Libraries

- Panda

Use for data manipulation and analysis and offer data structures and operations for manipulating numerical tables.

- Numpy

Use as an efficient multi-dimensional container of generic data.

- Pyodbc

An open-source python module which allow to access ODBC database.

- Mlxtend

A machine learning extension which provides functions for everyday data analysis and contain with functions such as counterfactual record creation, plot decision regions, drawing matrix of scatter plots and etc. (Alizadeh, 2020).

- Matplotlib

Provide an object-oriented API for embedding plots into applications.

- Scipy

Create numpy array object and part od Numpy stach that includes tools such as Matplotlib, pandas and Sympy.

- Ploty

A python graphing library that allows to make quality graphs with interactively. Example charts include such as scatter plots, area charts, bubble charts and etc.

- Dash

Is a python framework for building interactive web applications which build on top of flask, plotly.js, react and react.js (Mwiti, 2018).

3.6 Summary

This chapter summarizes the methodology which is used in order to achieve the objectives. For finding the frequent items and frequent paths, the FP growth algorithm was used. In discount calculation, customers are segmented based on RFM analysis by considering measures of total number of orders per customer, average days between orders per customer, and average size of orders per customer. Based on the results of the RFM analysis, customers have been divided into four clusters. Clustering has been used to identify customer behavior, and a discounted item list has been generated as a result. Software and libraries which are used in developing the project are also discussed here.

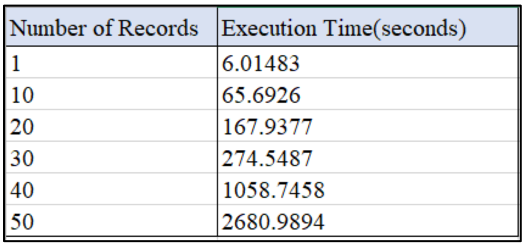
CHAPTER 4

EVALUATION AND RESULTS

This chapter presents the final results of the project and analysed the results using the appropriate statistical methods. Evaluation is discussed based on execution time of the FP growth algorithm over the number of results used, how the discount affects sales, and the effectiveness of the current layout over the previous layout.

4.1 Evaluate the execution time of the FP growth algorithm

The primary objective of the project was to find the most traveled path of the customer. In order to achieve this, the FP growth algorithm was used to identify the most traveled path of the selected customer. Because of the reasons in Table 3.2, the FP growth algorithm was selected for frequent path calculation in this project. In discussing the runtime for the FP growth algorithm, consider Figure 4.1.



Number of Records	Execution Time(seconds)
1	6.01483
10	65.6926
20	167.9377
30	274.5487
40	1058.7458
50	2680.9894

Figure 4.1: Time for execution of record/s

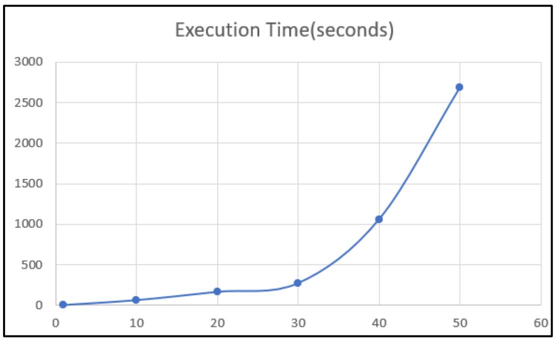
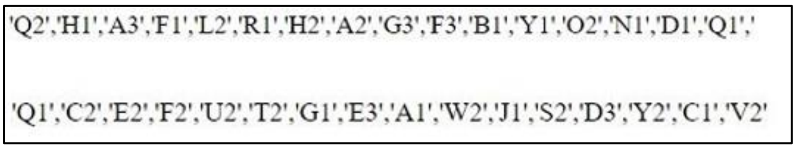


Figure 4.2: Time for execution of record/s graph

Figure 4.1 summarizes the execution time of FP growth to identify the frequent path for a different number of customers in seconds. Figure 4.1 Table data is graphically visualized in Figure 4.2. (Execution time may be different based on the performance of the computer.). As in Figure 4.2, when the number of records is increasing, execution time is also increased. Through this finding, FP growth can be identified as a timely and effective algorithm for frequent path mining in comparison to other algorithms.

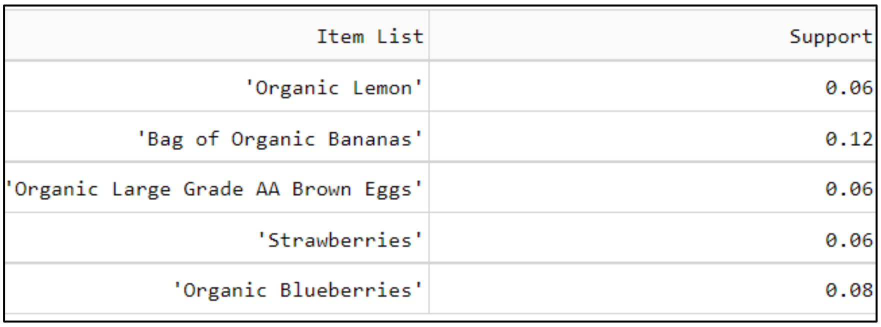
4.2 Improve sales based on discount schema.

Using the FP growth algorithm, it is possible to select the frequent path of a particular customer. This path indicates the location codes which the customer mostly travelled in the shop during the purchase. According to the customer code '516', his/her frequent path is in Figure 4.3 and the frequent item set is in Figure 4.4.



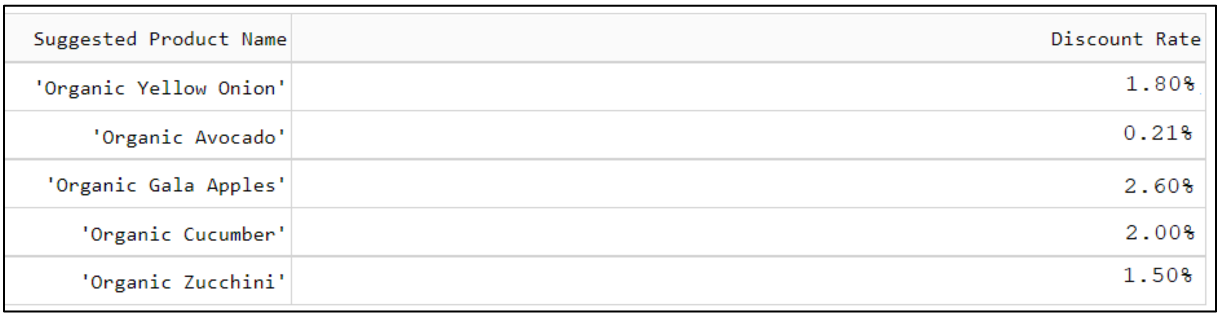
```
'Q2','H1','A3','F1','L2','R1','H2','A2','G3','F3','B1','Y1','O2','N1','D1','Q1','

'Q1','C2','E2','F2','U2','T2','G1','E3','A1','W2','J1','S2','D3','Y2','C1','V2'
```

Figure 4.3: Frequent path of customer '516'



Item List	Support
'Organic Lemon'	0.06
'Bag of Organic Bananas'	0.12
'Organic Large Grade AA Brown Eggs'	0.06
'Strawberries'	0.06
'Organic Blueberries'	0.08

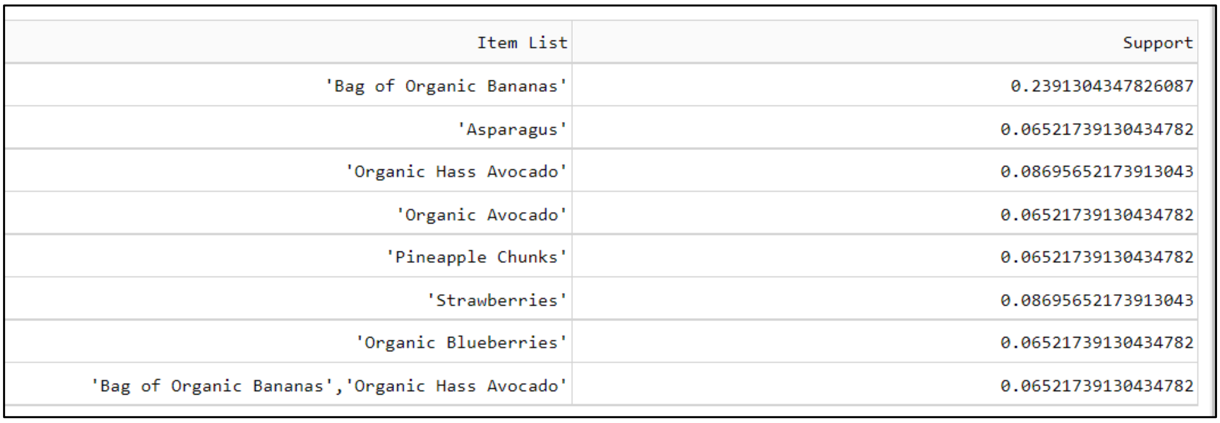
Figure 4.4: Frequent items of customer '516'



Suggested Product Name	Discount Rate
'Organic Yellow Onion'	1.80%
'Organic Avocado'	0.21%
'Organic Gala Apples'	2.60%
'Organic Cucumber'	2.00%
'Organic Zucchini'	1.50%

Figure 4.5: Discount applicable items of customer '516'

In this frequent path, there are locations that are travelled by the user, but some items are not bought. In order to solve this issue through this project, a discount is introduced for items which are bought by other customers. To motivate customers to buy a product, three discount schemas were introduced under section 3.4.1, and these may encourage customers to buy products which they have never bought or may be bought less frequently.



Item List	Support
'Bag of Organic Bananas'	0.2391304347826087
'Asparagus'	0.06521739130434782
'Organic Hass Avocado'	0.08695652173913043
'Organic Avocado'	0.06521739130434782
'Pineapple Chunks'	0.06521739130434782
'Strawberries'	0.08695652173913043
'Organic Blueberries'	0.06521739130434782
'Bag of Organic Bananas','Organic Hass Avocado'	0.06521739130434782

Figure 4.6: Frequent Items of customer '626'

Figure 4.6 contains the frequent items of customer '626', and one of the frequently bought item is 'Organic Avocado'. But that product is not a frequent item of customer '516' as stated in Figure 4.4. By introducing a discount for non-frequent items (Figure 4.5), customers may tend to buy that product on their next shopping trip. This led to an increase in sales of products and an increase in the revenue of the shop. 'Organic Gala Apples' is not an item bought by customer '516' but that is introduced with a discount to the customer. Based on these facts, we can conclude that customer '516' will buy these products with the influence of discount.

4.3 Effectiveness of Current layout over Previous Layout

In the previous section, a frequent path is identified. Customers are more likely to buy products once they are in the frequent path, and introducing discounts will also increase sales. Through this section it elaborates the effectiveness of the current layout over the previous layout.

The current layout is suggested based on the most frequent customers, which are identified in customer segments through RFM analysis. Through RFM analysis, four main customer segments have been identified, and they can be summarized as follows.

Cluster 0: Customers who place an average number of orders have an average visit rate but a large number of products in their orders.

Cluster 1: These are the customers who place the lowest order rates but do visit the shop often and, once visited, place an average number of products in their order.

Cluster 2: This segment of customers places more orders, and as they often visit the shop, each order contains an average number of products.

Cluster 3: Customers in this segment can be identified as the most order placing segment but not as frequent with the least number of products in their orders when compared to the other segments.

Following Figure 4.7 illustrate overall behavior of the clusters based on the travel frequency of each location inside the supermarket. Based on this graph can conclude that each cluster contain with parallel behavior.

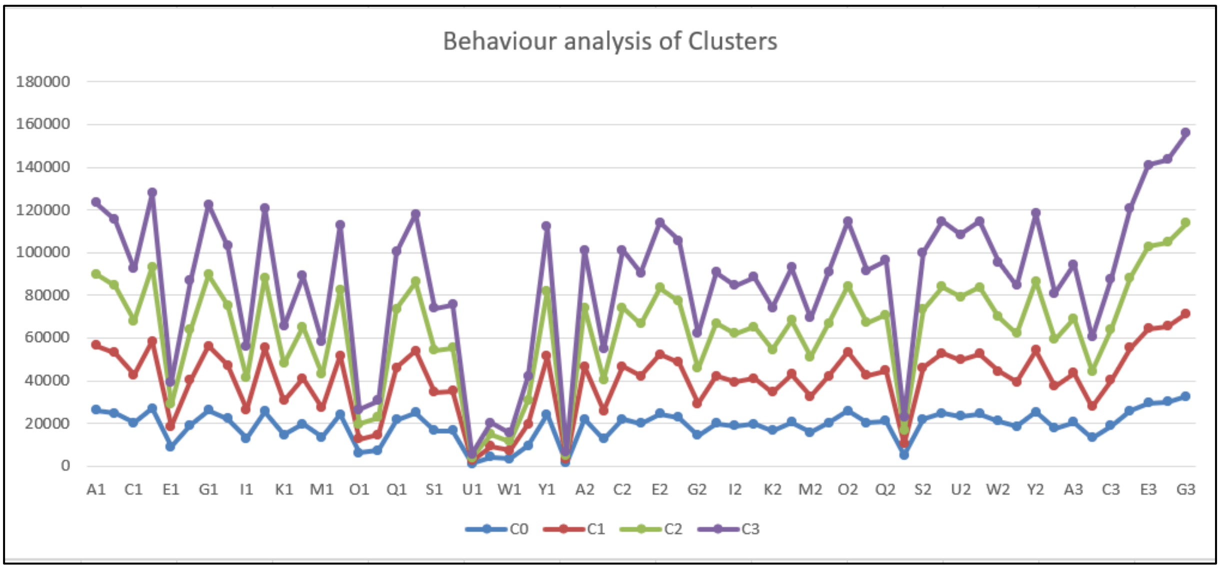


Figure 4.7: Behavior analysis of cluster

Cluster two contain with the customers who frequently visit the shop according to the RFM analysis. As because of above reasons generation of the proposed layout, cluster two data is used. Previous layout of the shop is visualized in the Figure 4.7.

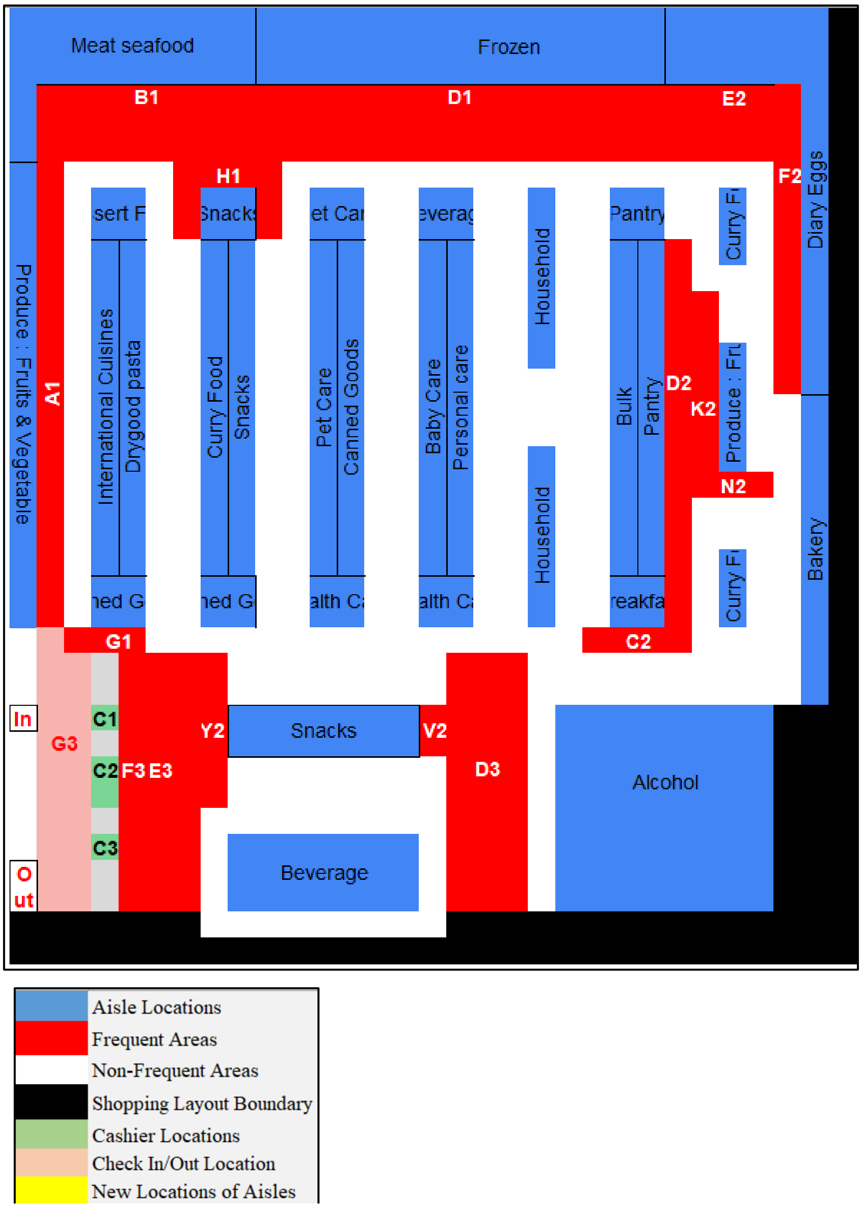
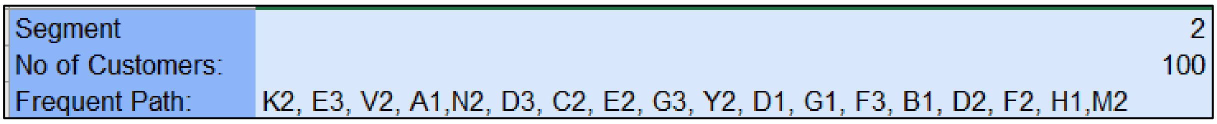


Figure 4.8: Previous Layout



Segment	2
No of Customers:	100
Frequent Path:	K2, E3, V2, A1,N2, D3, C2, E2, G3, Y2, D1, G1, F3, B1, D2, F2, H1,M2

Figure 4.9: Summery relate to frequent path of Figure 4.8

Based on a hundred customers, segment two customers were used to identify the frequent path, and the FP growth algorithm was used with a support value of 0.94. The frequent path which was identified through the FP growth algorithm is shown in Figure 4.9. Most of the customers have traveled around the shopping aisles but not inside the shop. Figure 4.10 illustrates the sales of the products in each area.

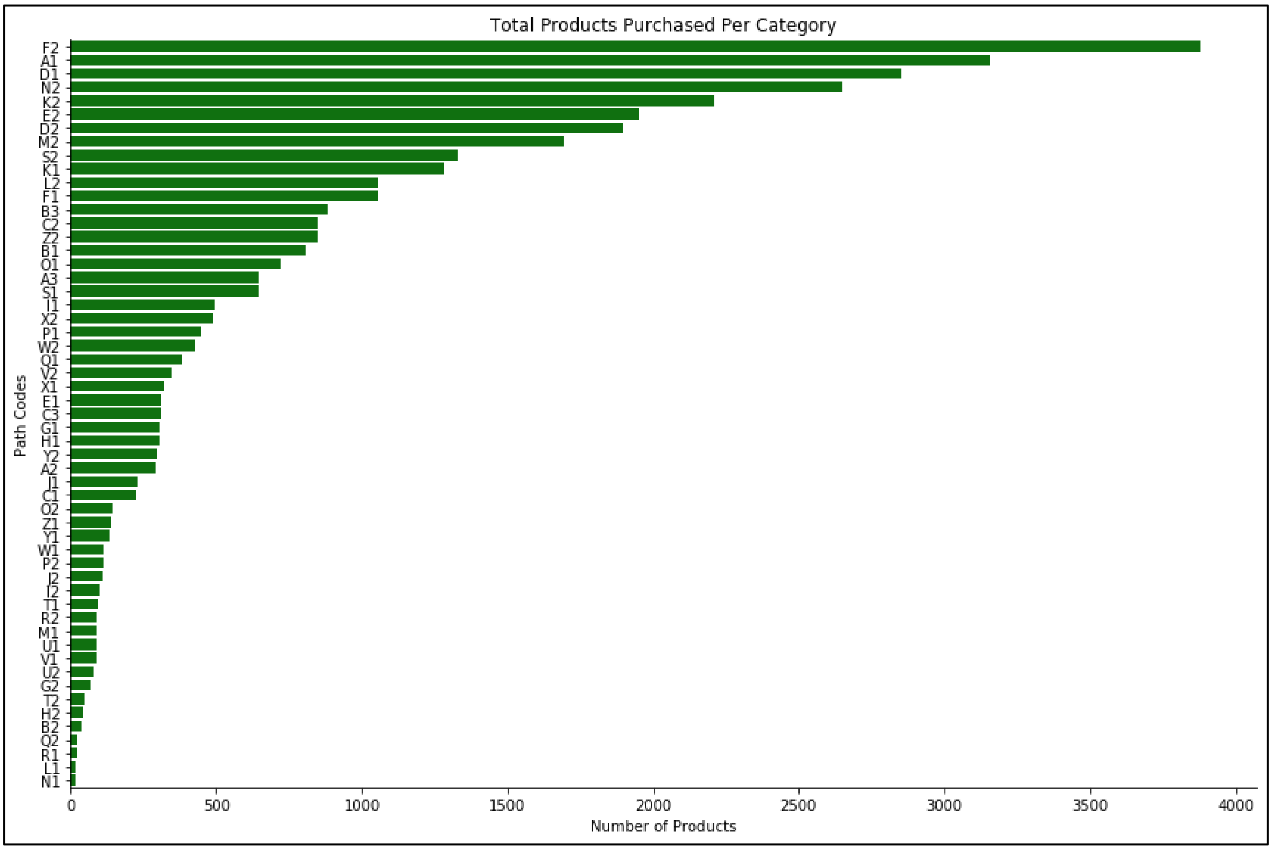


Figure 4.10: Total products purchased per Path

According to Figure 4.10, a significant number of products which are purchased by customers are located in the frequent path. In comparing the frequent path codes to the above graph, we can identify that some of the frequent paths don’t contain a considerable number of purchased products. The reason for this is that even though customers travel to some locations, they may not tend to purchase many items from those locations, such as products near the cashier counter, entrance or alcohol area.

In order to increase sales, section 3.4.1 introduced three different discount schemas. Using these will improve sales, as well as changing the layout will affect sales in parallel.

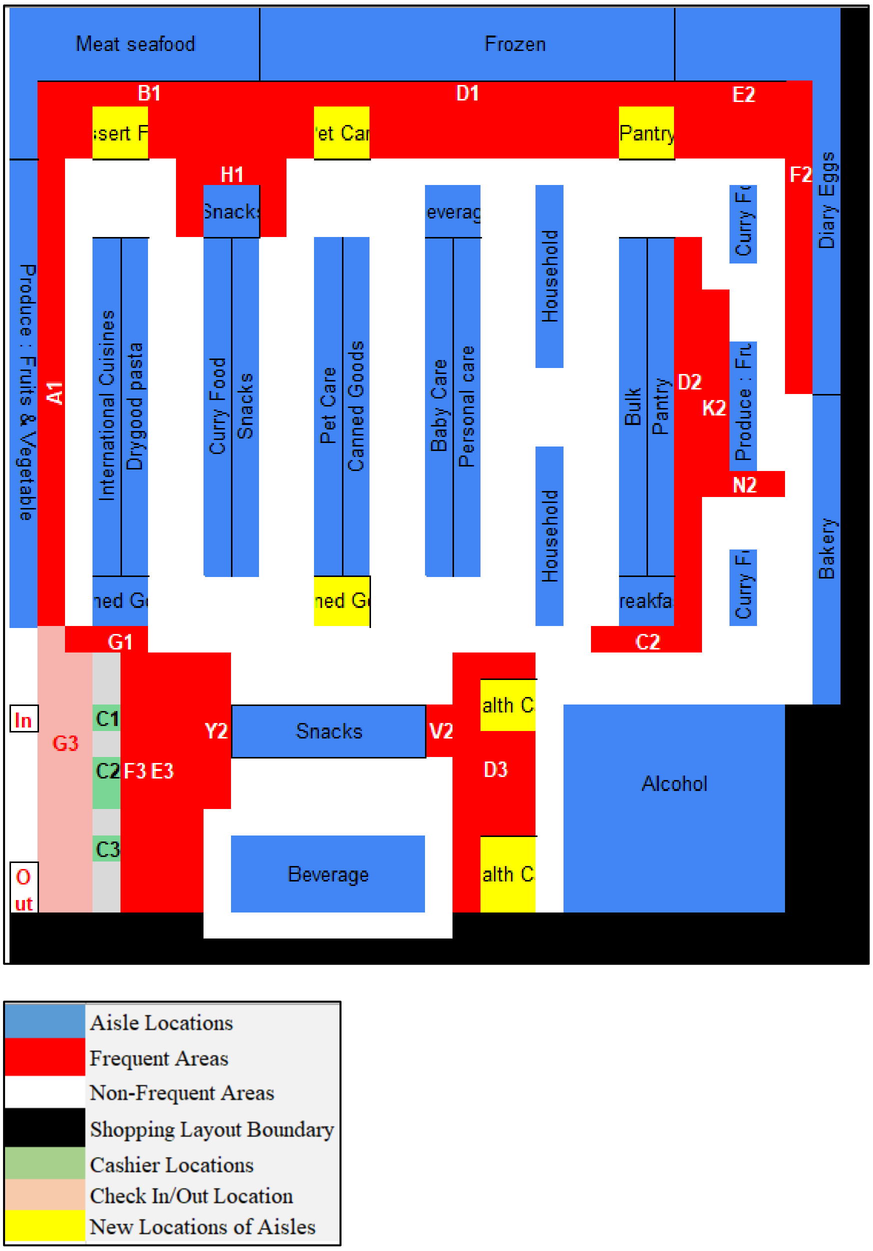


Figure 4.11: Current Layout

Placement of the products in the frequent travel path, which has fewer sales, will result in an increase in the sales of the shop. Figure 4.11 illustrates the current layout of the shop, which changes some of the aisle locations in order to increase sales. Even though some products are bought by the customers, they are not located in their frequent path. However, according to the current layout, they are placed in frequent paths, which may assist customers in easily locating the products, thereby increasing sales. Changes in the layout affect mostly those customers who visit more often. But introducing a discount schema will improve the sales of irregular customers, and they will tend to visit the shop more often.

4.4 Summary

Through this chapter, it discussed the findings of the project and evaluated them based on different factors. The FP growth algorithm was used as it contains significant advantages over other association rule mining techniques. Introducing discounts for less frequent items and non-purchase items is a comprehensive way of marketing in order to increase sales. Placing the non-frequent items on a frequent path is another mechanism for increasing sales. Also, updating the layout of the shop based on the travel path is a comprehensive way to attract customers.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This project proposed a new discount schema and new supermarket layout based on the association among the traveled path of customers inside supermarket by applying FP growth algorithm and RFM analysis. These techniques allow retailers to understand the consumers travel habits, purchase habits and then create a strong demand for current busy consumers. Using of the data mining provide retailer to understand what consumer want and approaches to achieve that while getting profit and satisfying the customer also.

FP growth algorithm is an improved version of the Apriori algorithm it used for frequent pattern mining from the user data. Results are computed based on the supermarket transaction data and it indicate the real behavior of the consumer. FP growth algorithm was used to identify the frequent path of each customer and based on that it allowed to identify the frequent areas consumers mostly traveled and non-frequent areas also. Based on FP growth algorithm identified the most frequent items of a customer and non-frequent items which are not bought by the customer but others. In order to increase the sales of the non-frequent items discount schemas were introduced. As the discount schemas are based on the consumer travel pattern these will have positive impact on the revenue of the retailers. Discount schemas are personalized to the consumer and they are primarily focus to create a strong connection between consumer and retailer.

As the dataset contain with large transaction data with different customers RFM analysis was used as a customer segmentation method to group the customers based on their similarities with respect to different dimensions. Using of the RFM analysis allowed to identify the consumer segments and RFM analysis is quantitively method which allow to identify customer segments based on Recency, Frequency and Monitory value. Based on a fraction of users identified the most frequent travel areas and non-frequent travel areas. Introducing a new shopping layout with relocating the aisles in non-frequent areas to frequent travel areas will facilitate the consumer to find their products easily and also to increase the sales simultaneously.

Providing a discount based on customer travel path will improve the sales rather than a regular discount for every customer. Finally, can conclude that recommending discount and shopping layout based on travel path of customer will result to increase the sales and improve the customer satisfaction of the shop.

5.1 Future work

Implementing an advanced pattern mining algorithm in parallel to the FP growth algorithm will improve performance of result generation. For an accurate result generation in path tracking, it is needed to track the location correctly. As a further development of this project for location tracking can use indoor location tracking techniques.

Using of the RFM analysis allowed to identify the customer segments, for these segments can use other data mining technologies such as churn prediction to identify customers who will likely to cancel the shopping in future and introduced promotions for them.

As the suggested discount schemas are for the product categories, they can further personalize to item wise for a customer and for can use effective discount schema for discount calculation.

REFERENCES

[1]. Chen, Y. L., Kuo, M., Wu, S., & Tang, K. (2009). Discovering recency, frequency, and monetary (RFM) sequential patterns. *Electronic Commerce Research and Applications*, 241-251.

[2]. Levav, J., & Zhu, R. J. (2009). Seeking freedom through variety. *Journal of Consumer Research, 36*, 600-610.

[3]. Aggelis, V., & Christodoulakis, D. (2005). Customer Clustering using RFM analysis.

[4]. Alizadeh, E. (2020). *MLxtend: A Library with Interesting Tools for Data Science Tasks*. Retrieved 05 28, 2021, from https://towardsdatascience.com/mlxtend-a-python-library-with-interesting-tools-for-data-science-tasks-d54c723f89cd

[5]. Alyoubi, K. H. (2020). Association Rule Mining on Customer's Data using Frequent Pattern Algorithm. *IJCSNS International Journal of Computer Science and Network Security, 20*(5).

[6]. Ansari, A., & Riasi, A. (2016). Taxonomy of Marketing Strategies Using Bank Customers' Clustering. *International Journal of Business and Management, 11*(7).

[7]. Chandon, P., Hutchinson, J. W., & Bradlow, E. T. (2009). Does in-store marketing work? Effects of the number and position of shelf facings on brand attention and evaluation at the point of purchase. *... of Marketing, 73*, 1-17.

[8]. Chee, C. H., Jaafar, J., Aziz, I. A., Hasan, M. H., & Yeoh, W. (2019). Algorithms for frequent itemset mining: a literature. *Artif Intell Rev*, 2603-2621.

[9]. Coussement, K., Van den Bossche, F., & De Bock, K. (2014). Data accuracy's impact on segmentation performance: Benchmarking RFM analysis, logistic regression, and decision trees. *Journal of Business Research, 67*(1), 2751-2758.

[10]. Drèze, X., Hoch, S. J., & Purk, M. E. (1994). Shelf management and space elasticity. *Journal of Retailing, 70*(4), 301-326.

[11]. Elbers, T. (2016). The effects of in-store layout - and shelf designs on consumer behaviour.

[12]. Gurudath, S. (2020). Market Basket Analysis & Recommendation System Using Association Rules.

[13]. Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. *ACM SIGMOD Record, 29*(2), 1-12.

[14]. Hu, Y., & Yeh, T. W. (2014). Discovering valuable frequent patterns based on RFM analysis without customer identification information. *Knowledge-Based Systems, 61*, 76-88.

[15]. Jonker, J. J., Piersma, N., & Poel, D. (2004). Joint optimization of customer segmentation and marketing policy to maximize long-term profitability. *Expert Systems with Applications, 27*(2), 159-168.

[16]. Larson, J. S., Bradlow, E., & Fader, P. S. (2005). An exploratory look at supermarket shopping paths. *International Journal of Research in Marketing, 22*(4), 395-414.

[17]. Lewison, D. M. (1996). *Retailing.* Toledo, OH,USA: Prentice Hall.

[18]. Lim, T. P. (2019). *The Most Important Data Science Tool for Market and Customer Segmentation*. Retrieved 05 12, 2021, from https://towardsdatascience.com/the-most-important-data-science-tool-for-market-and-customer-segmentation-c9709ca0b64a

[19]. Liu, S. S., Melara, R., & Arangarasan, R. (2007). The Effects of Store Layout on Consumer Buying Behavioral Parameters with Visual Technology. *Journal of Shopping Center Research Journal of Shopping Center Research, 14*, 63-72.

[20]. Mwiti, D. (2018). *Dash for Beginners*. Retrieved 05 03, 2021, from https://www.datacamp.com/community/tutorials/learn-build-dash-python

[21]. Sag˘lam, B., Salman, F. B., Sayın, S., & Tu¨rkay, M. (2006). A mixed-integer programming approach to the clustering problem with an application in customer segmentation. *European Journal of Operational Research*, 866-879.

[22]. Smulders, J. (2019). Clustering customer trajectories using Operational Edit Distance.

[23]. Sorensen, H. (2003). The Science of Shopping. *Marketing Research, 15,* 30-35.

[24]. Soudagar, R. (2012). Customer Segmentation and Strategy Definition in Segments.

[25]. Valenzuela , A., & Raghubir , P. (2009). Center of Orientation: Effect of Vertical and Horizontal Shelf Space Product Position. *Association for Consumer Research, 36,* 100-103.

[26]. Valenzuela, A., Raghubir, P., & Mitakakisa, C. (2013). Shelf space schemas: Myth or reality? *Journal of Business Research, 66*(7), 881-888.

[27]. Yen, S. W., Wang, C. H., & Ouyang, L. Y. (2012). A Search Space Reduced Algorithm for Mining Frequent Patterns. *Journal of Information Science and Engineering , 28*, 177-191.

APPENDICES

Appendix A – Source code

Class for frequent item selection and discount calculation

```
import pandas as pd
import numpy as np
import sys
import pyodbc
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules
import pyodbc

itemlist= ""

def myencoder(i):
    if i <= 0:
        return 0
    elif i>= 1:
        return 1

def getCustomerfrequentitem(user_id, pathlist):
    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql= f"""select Order_D.order_id , Order_D.product_id  from Order_D
join products on products.product_id = Order_D.product_id  where user_id =
{user_id} and PathCode in ({pathlist}) """
    item_list_df = pd.read_sql(sql,con)

    sql2= f"""select product_id,product_name from products """
    item_df = pd.read_sql(sql2,con)
    con.close()

    item_list_df['temp']=1

item_list_df.rename(columns={'order_id':'order','product_id':'product'},inp
lace=True)
    df_encoder =
item_list_df.groupby(['order','product'])['temp'].sum().unstack().fillna(0)

    df_encoder = df_encoder.applymap(myencoder)
    frequent_itemsets = fpgrowth(df_encoder,
min_support=0.05,use_colnames=True)

    frequent_itemsets["itemsets"] =
frequent_itemsets["itemsets"].apply(lambda x: ',
'.join(list(x))).astype("unicode")
    frequent_itemsets['baseitem'] = None

    for index,row in frequent_itemsets.iterrows():
        totalitemlist =""
        for item in (str(row['itemsets'])).split(","):
            product_id = int(item)
            df =item_df.loc[item_df.product_id ==
product_id][['product_name']]
            totalitemlist = totalitemlist + ',' +
str(df.product_name.values).strip('[]')
        frequent_itemsets.at[index,'itemsets'] = totalitemlist.lstrip(',')
        frequent_itemsets.at[index,'baseitem'] = str(row['itemsets'])

    frequent_itemsets = frequent_itemsets[['itemsets',
'support','baseitem']]
    frequent_itemsets.rename(columns = {'itemsets':'Item List',
'support':'Support'}, inplace = True)
    return frequent_itemsets

def getCustomerdiscountRate(id_user,id_frequentitemlist,id_frequentpath):
    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql2= f"""SELECT antecedents,consequents  FROM Item_Rules where cluster
= (select Cluster from User_ByCluster where user_id = {id_user}) """
    itemRule_df = pd.read_sql(sql2,con)
    sql3= f"""select
product_id,product_name,PathCode,Cost,CashFlow,ExpectedGrowth from products
"""
    item_df = pd.read_sql(sql3,con)
    con.close()

    discount_df = pd.DataFrame(columns=['Suggested Product Name', 'Discount
Rate'])

    for i in id_frequentitemlist.split(","):
        abc = "".join(filter(str.isdigit, i))
        values = ((itemRule_df.loc[itemRule_df['antecedents'] ==
abc]).groupby('antecedents')['consequents'].apply(','.join).values)
        if values.size > 0:
            for item in (str(values)).split(","):
                numeric_string = "".join(filter(str.isdigit, item))
                itemdetail_df = item_df[(item_df.product_id ==
int(numeric_string))]
                product_name = ((itemdetail_df.product_name).values)

pathcode=(str(str((itemdetail_df.PathCode).values)).strip('[]').replace("'"
, ""))
                CashFlow = (itemdetail_df.CashFlow.values)
                ExpectedGrowth = (itemdetail_df.ExpectedGrowth)
                discountRate =
str(round(float((pow((ExpectedGrowth/CashFlow),1) - 1).values), 3))
                if pathcode in id_frequentpath:
                    discount_df = discount_df.append({'Suggested Product
Name': str(product_name).strip('[]'),'Discount
Rate':str(discountRate).strip('[]')}, ignore_index=True)
    discount_df.drop_duplicates(subset ="Suggested Product Name",keep =
'first', inplace = True)
    return discount_df

def getPredictMeanItemList(id_department):
    con  = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
```

```
    sql2 = f"""SELECT  * from products where department_id =
{id_department} """
    itemlist_df = pd.read_sql(sql2,con)

    df = itemlist_df[itemlist_df['itempercentage'] <
itemlist_df.itempercentage.mean()]
    df = df[['product_name','itempercentage']]
    df.sort_values(by='itempercentage', ascending=False)
    return df

def getPredictMeanBinList(id_department):
    con  = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql2 = f"""SELECT  * from products where department_id =
{id_department} """
    itemlist_df = pd.read_sql(sql2,con)
    df2 = itemlist_df[itemlist_df['itempercentage'] >
itemlist_df.itempercentage.mean()]
    df_Binlist = df2[['BinNumber']]
    binlist = "'" + df_Binlist['BinNumber'].str.cat(sep="', '") + "'"

    binlist = ','.join(set(binlist.replace("", "").split(',')))
    sql= f"""SELECT distinct consequents as Bin_Location_Code FROM
BinRulesbyDepartment where antecedents in ({binlist})  and department =
{id_department}"""
    bin_list_df = pd.read_sql(sql,con)
    con.close()
    bin_list_df.sort_values("Bin_Location_Code", inplace = True)
    bin_list_df.drop_duplicates(subset ="Bin_Location_Code", keep = False,
inplace = True)


    return bin_list_df
```

Class for customer frequent path identification

```
import pandas as pd
import numpy as np
import sys
import pyodbc
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules

def myencoder(i):
    if i <= 0:
        return 0
    elif i>= 1:
        return 1

def getCustomerfrequentpath(user_id):
    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql = "SELECT order_id ,user_id, Path FROM ORDERS "
    pathbyuser_df = pd.read_sql(sql,con)
    con.close()
    pathbyuser_df = pathbyuser_df[pathbyuser_df['user_id'] == str(user_id)]
    for index,row in pathbyuser_df.iterrows():
        pathbyuser_df.at[index,'Path'] = str([row.Path[i:i+2] for i in
range(0, len(row.Path), 2)])

    nw_pathbyuser_df =
pd.DataFrame(pathbyuser_df.Path.str.split(',').tolist(),
index=pathbyuser_df.order_id).stack()
    nw_pathbyuser_df = nw_pathbyuser_df.reset_index([0, 'order_id'])
    nw_pathbyuser_df.columns = ['order_id', 'Path']
    nw_pathbyuser_df['Path'] =
nw_pathbyuser_df['Path'].str.strip('[]').astype(str)

    nw_pathbyuser_df['temp']=1

nw_pathbyuser_df.rename(columns={'order_id':'order','Path':'Path'},inplace=
True)

    df_encoder =
nw_pathbyuser_df.groupby(['order','Path'])['temp'].sum().unstack().fillna(0
)
    df_encoder = df_encoder.applymap(myencoder)
    frequent_itemsets = fpgrowth(df_encoder,
min_support=0.7,use_colnames=True)

    frequent_itemsets["itemsets"] =
frequent_itemsets["itemsets"].apply(lambda x:
','.join(list(x))).astype("unicode")
    list_frequentvalue = (",".join(frequent_itemsets['itemsets'].tolist()))
    frequent_locations =  ','.join(set(list_frequentvalue.replace(" ",
"").split(',')))

    return frequent_locations
```

Class for Heatmap Generation

```
import random
import pandas as pd
import scipy.sparse as sparse
import numpy as np
import datetime as dt
import pyodbc
import collections
import os
from matplotlib.patches import Rectangle

#df_result = pd.read_csv('E:/My
Academic/MSc/Project/2ndTime/ExcelData/result3.csv')
order_wth_path_df=[]

def methodGetuseridlist():
    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql = "select user_id from TopCustomerList order by user_id asc"
    dfuseridlist= pd.read_sql(sql,con)
    con.close()
    return dfuseridlist

def methodGetdepartmentidlist():
    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql = "select department_id , department  from departments order by
department_id asc"
    dfdepartmentidlist= pd.read_sql(sql,con)
    con.close()
    return dfdepartmentidlist

def methodintialGetTravelpathcount(user_id):

    df_shoppingcart = pd.read_csv('E:/My
Academic/MSc/Project/2ndTime/ExcelData/shopping cart
navigation_AXISCOUNT.csv')

    con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")
    sql = "SELECT order_id,user_id,order_dow,Path FROM
[DBRecommendation].[dbo].[orders] where user_id in (select user_id from
TopCustomerList)"
    order_wth_path_df = pd.read_sql(sql,con)
    con.close()

   # order_wth_path_filtered =
order_wth_path_df[order_wth_path_df['order_dow'] == order_dwo]
    order_wth_path_nwfiltered =
order_wth_path_df[order_wth_path_df['user_id'] == str(user_id)]


    str_count0 = 0
    str_count1 = 0
    str_count2 = 0
    str_count3 = 0
    str_count4 = 0
    str_count5 = 0
    str_count6 = 0
    str_count7 = 0
    str_count8 = 0
    str_count9 = 0
    str_count10 = 0
    str_count11= 0
    str_count12 = 0
    str_count13 = 0
    str_count14 = 0
    str_count15 = 0
    str_count16 = 0
    str_count17 = 0
    str_count18 = 0
    str_count19 = 0
    str_count20 = 0
    str_count21 = 0
    str_count22= 0
    str_count23 = 0
```

```
str_count24 = 0
str_count25 = 0
str_count26 = 0

str_count27 = 0
str_count28 = 0
str_count29 = 0
str_count30 = 0
str_count31 = 0
str_count32 = 0
str_count33 = 0
str_count34 = 0
str_count35 = 0
str_count36 = 0
str_count37 = 0
str_count38 = 0
str_count39 = 0
str_count40 = 0
str_count41 = 0
str_count42 = 0
str_count43 = 0
str_count44 = 0
str_count45 = 0
str_count46 = 0
str_count47 = 0
str_count48 = 0
str_count49 = 0
str_count50 = 0
str_count51 = 0
str_count52 = 0
str_count53 = 0
str_count54 = 0
str_count55 = 0
str_count56 = 0
str_count57 = 0
str_count58 = 0
str_count59 = 0

for index, row in order_wth_path_nwfiltered.iterrows():
    # access data using column names
    str1 = row['Path']
    str_count1 = str_count1 + str1.count('A1')
    str_count2 = str_count2 + str1.count('B1')
    str_count3 = str_count3 + str1.count('C1')
    str_count4 = str_count4 + str1.count('D1')
    str_count5 = str_count5 + str1.count('E1')
    str_count6 = str_count6 + str1.count('F1')
    str_count7 = str_count7 + str1.count('G1')
    str_count8 = str_count8 + str1.count('H1')
    str_count9 = str_count9 + str1.count('I1')
    str_count10 = str_count10 + str1.count('J1')
    str_count11 = str_count11 + str1.count('K1')
    str_count12 = str_count12 + str1.count('L1')
    str_count13 = str_count13 + str1.count('M1')
    str_count14 = str_count14 + str1.count('N1')
    str_count15 = str_count15 + str1.count('O1')
    str_count16 = str_count16 + str1.count('P1')
    str_count17= str_count17 + str1.count('Q1')
    str_count18 = str_count18 + str1.count('R1')
    str_count19 = str_count19 + str1.count('S1')
    str_count20 = str_count20 + str1.count('T1')
    str_count21 = str_count21 + str1.count('U1')
    str_count22 = str_count22 + str1.count('V1')
    str_count23 = str_count23 + str1.count('W1')
    str_count24 = str_count24 + str1.count('X1')
    str_count25 = str_count25 + str1.count('Y1')
    str_count26 = str_count26 + str1.count('Z1')

    str_count27 = str_count27 + str1.count('A2')
    str_count28 = str_count28 + str1.count('B2')
    str_count29 = str_count27 + str1.count('C2')
    str_count30 = str_count30 + str1.count('D2')
    str_count31 = str_count31 + str1.count('E2')
    str_count32 = str_count32 + str1.count('F2')
    str_count33 = str_count33 + str1.count('G2')
    str_count34 = str_count34 + str1.count('H2')
    str_count35 = str_count35 + str1.count('I2')
    str_count36 = str_count36 + str1.count('J2')
    str_count37 = str_count37 + str1.count('K2')
    str_count38 = str_count38 + str1.count('L2')
    str_count39 = str_count39 + str1.count('M2')
    str_count40 = str_count40 + str1.count('N2')
    str_count41 = str_count41 + str1.count('O2')
    str_count42 = str_count42 + str1.count('P2')
    str_count43 = str_count43 + str1.count('Q2')
    str_count44 = str_count44 + str1.count('R2')
    str_count45 = str_count45 + str1.count('S2')
    str_count46 = str_count46 + str1.count('T2')
    str_count47 = str_count47 + str1.count('U2')
    str_count48 = str_count48 + str1.count('V2')
    str_count49 = str_count49 + str1.count('W2')
    str_count50 = str_count50 + str1.count('X2')
    str_count51 = str_count51 + str1.count('Y2')
    str_count52 = str_count52 + str1.count('Z2')
    str_count53 = str_count53 + str1.count('A3')
    str_count54 = str_count54 + str1.count('B3')
    str_count55 = str_count55 + str1.count('C3')
    str_count56 = str_count56 + str1.count('D3')
    str_count57 = str_count57 + str1.count('E3')
    str_count58 = str_count58 + str1.count('F3')
    str_count59 = str_count59 + str1.count('G3')

countval_list = collections.defaultdict(int)
countval_list['X'] = str_count0
countval_list['A1'] = (str_count1)
countval_list['B1']=(str_count2)
countval_list['C1']=(str_count3)
countval_list['D1']=(str_count4)
countval_list['E1']=(str_count5)
countval_list['F1']=(str_count6)
countval_list['G1']=(str_count7)
countval_list['H1']=(str_count8)
countval_list['I1']=(str_count9)
countval_list['J1']=(str_count10)
```

```
    countval_list['K1']=(str_count11)
    countval_list['L1']=(str_count12)
    countval_list['M1']=(str_count13)
    countval_list['N1']=(str_count14)
    countval_list['O1']=(str_count15)
    countval_list['P1']=(str_count16)
    countval_list['Q1']=(str_count17)
    countval_list['R1']=(str_count18)
    countval_list['S1']=(str_count19)
    countval_list['T1']=(str_count20)
    countval_list['U1']=(str_count21)
    countval_list['V1']=(str_count22)
    countval_list['W1']=(str_count23)
    countval_list['X1']=(str_count24)
    countval_list['Y1']=(str_count25)
    countval_list['Z1']=(str_count26)
    countval_list['A2']=(str_count27)
    countval_list['B2']=(str_count28)
    countval_list['C2']=(str_count29)
    countval_list['D2']=(str_count30)
    countval_list['E2']=(str_count31)
    countval_list['F2']=(str_count32)
    countval_list['G2']=(str_count33)
    countval_list['H2']=(str_count34)
    countval_list['I2']=(str_count35)
    countval_list['J2']=(str_count36)
    countval_list['K2']=(str_count37)
    countval_list['L2']=(str_count38)
    countval_list['M2']=(str_count39)
    countval_list['N2']=(str_count40)
    countval_list['O2']=(str_count41)
    countval_list['P2']=(str_count42)
    countval_list['Q2']=(str_count43)
    countval_list['R2']=(str_count44)
    countval_list['S2']=(str_count45)
    countval_list['T2']=(str_count46)
    countval_list['U2']=(str_count47)
    countval_list['V2']=(str_count48)
    countval_list['W2']=(str_count49)
    countval_list['X2']=(str_count50)
    countval_list['Y2']=(str_count51)
    countval_list['Z2']=(str_count52)
    countval_list['A3']=(str_count53)
    countval_list['B3']=(str_count54)
    countval_list['C3']=(str_count55)
    countval_list['D3']=(str_count56)
    countval_list['E3']=(str_count57)
    countval_list['F3']=(str_count58)
    countval_list['G3']=(str_count59)

    print("2 part")
    df_shoppingcartVV = df_shoppingcart
    resultdistinctC_list=[]
    df_list =[]
    character_list=
['X','A1','B1','C1','D1','E1','F1','G1','H1','I1','J1','K1','L1','M1','N1',
'O1','P1','Q1','R1','S1','T1','U1','V1','W1','X1','Y1','Z1','A2','B2','C2',
'D2','E2','F2','G2','H2','I2','J2','K2','L2','M2','N2','O2','P2','Q2','R2',
'S2','T2','U2','V2','W2','X2','Y2','Z2','A3','B3','C3','D3','E3','F3','G3']
    column_list
=['c1','c2','c3','c4','c5','c6','c7','c8','c9','c10','c11','c12','c13','c14
','c15','c16','c17','c18','c19','c20','c21','c22','c23','c24','c25','c26','
c27','c28','c29','c30','c31']
    for val in column_list:
        values = df_shoppingcart[val].unique()
        resultdistinctC_list.append(values)

    number = 0
    print("resultdistinctC_list")

    for val in column_list:
        values= resultdistinctC_list[number]
        for val2 in values:
            val3 = df_shoppingcart[df_shoppingcart[val] ==
val2].index.tolist()
            anser = (val,val2,val3)
            df_list.append(anser)
        number = number +1
    print("valuee")

    for row in df_list:
        for valrow in row[2]:
            for key, value in countval_list.items():
                if key == row[1]:
                    df_shoppingcartVV.loc[valrow,row[0]] = value

    df_shoppingcartVV.to_excel(r'result3.xlsx', index = False)

    if os.path.exists('result3.csv'):
        os.remove('result3.csv')
        print("file deleted")
    else:
        print("No such file")

    df_shoppingcartVV.to_csv('result3.csv', encoding='utf-8', index=False)
    df_newresult = pd.read_csv('result3.csv')

    return df_newresult
```

Class for RFM analysis

```
# Import needed packages

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns
```

```
sns.set_context('talk')

sns.set_style('white')

import sqlite3

import scipy.stats as stats

#from welch_functions import *

from statsmodels.stats.power import TTestIndPower, TTestPower

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

from sklearn.manifold import TSNE

from sklearn.metrics import silhouette_score

from yellowbrick.cluster import KElbowVisualizer

import pyodbc



con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")

sql="SELECT  * FROM [DBRecommendation].[dbo].[Orders] "

orders = pd.read_sql(sql,con)

con.close()



con = pyodbc.connect("DRIVER={SQL
Server};server=localhost;database=DBRecommendation")

sql="SELECT order_id, COUNT(product_id) AS num_products    FROM Order_D
GROUP BY order_id "

prod_counts = pd.read_sql(sql,con)

con.close()



prod_counts.set_index('order_id', inplace=True)

user_data = orders.join(prod_counts, how='inner', on='order_id')
```

```
num_orders = user_data.groupby('user_id')['order_number'].max()
user_data['order_on_peak'] = np.where(user_data['order_dow'] <=1, 1, 0)
peakday_rate = round(user_data.groupby('user_id')['order_on_peak'].mean(), 2)
med_hour = round(user_data.groupby('user_id')['order_hour_of_day'].median(), 0)

user_data['peak_time'] = np.where((user_data.order_hour_of_day >= 10)
                                  & (user_data.order_hour_of_day <= 16), 1, 0)
peaktime_rate = round(user_data.groupby('user_id')['peak_time'].mean(), 2)
mean_lag = round(user_data.groupby('user_id')['days_since_prior_order'].mean(), 0)
mean_products = round(user_data.groupby('user_id')['num_products'].mean(), 0)

features = pd.concat([num_orders, peakday_rate, med_hour, peaktime_rate,
                      mean_lag, mean_products], axis=1)
features.columns = ['num_orders', 'peakday_rate', 'med_hour', 'peaktime_rate',
                    'mean_lag', 'mean_products']

features['num_orders'] = np.log(features['num_orders'])

scaler = StandardScaler()
feat_scaled = scaler.fit_transform(features)

model = KMeans()
visualizer = KElbowVisualizer(model, k=(2, 21))

def optimal_kmeans(dataset, start=2, end=11):
    n_clu = []
    km_ss = []
    for n_clusters in range(start, end):
        kmeans = KMeans(n_clusters=n_clusters)
        labels = kmeans.fit_predict(dataset)
        silhouette_avg = round(silhouette_score(dataset, labels,
                                                random_state=1), 3)
        km_ss.append(silhouette_avg)
        n_clu.append(n_clusters)

        print("No. Clusters: {}, Silhouette Score: {}, Change from Previous Cluster: {}".format(
            n_clusters,
            silhouette_avg,
            (km_ss[n_clusters - start] - km_ss[n_clusters - start - 1]).round(3)))
        if n_clusters == end - 1:
            plt.figure(figsize=(4,4))

            plt.title('Silhouette Score Elbow for KMeans Clustering')
            plt.xlabel('k')
            plt.ylabel('silhouette score')
            sns.pointplot(x=n_clu, y=km_ss)
            plt.savefig('silhouette_score.png', format='png', dpi=300,
                        pad_inches=2.0)
            plt.tight_layout()
            plt.show()

def kmeans(df, clusters_number):
    kmeans = KMeans(n_clusters = clusters_number, random_state = 1)
    kmeans.fit(df)
    cluster_labels = kmeans.labels_
    df_new = df.assign(Cluster = cluster_labels)
    model = TSNE(random_state=1)
    transformed = model.fit_transform(df)
    plt.title('Flattened Graph of {} Clusters'.format(clusters_number))
    sns.scatterplot(x=transformed[:,0], y=transformed[:,1],
                    hue=cluster_labels, style=cluster_labels, palette="Set1")

    return df_new, cluster_labels

scaler = StandardScaler()
feat_few_scaled = scaler.fit_transform(features_fewer)

feat_few_scaled = pd.DataFrame(feat_few_scaled)

cluster_melt = pd.melt(cluster_df.reset_index(),
                       id_vars=['user_id', 'Cluster'],
                       value_vars=['Number of Orders',
                                   'Avg. Lag Between Orders',
                                   'Avg. # Products Per Order'],
                       var_name='Metric',
                       value_name='Value')

palette = ['lightgreen', 'orange', 'steelblue','yellow']
plt.figure(figsize=(10,5))
sns.pointplot(x='Metric', y='Value', data=cluster_melt, hue='Cluster',
          palette=palette)
plt.xlabel(")
plt.ylabel('Value')
plt.yticks([])
plt.title('Three Customer Segments')
sns.despine()
plt.tight_layout()
plt.savefig('snake_plot.png', dpi=300, pad_inches=2.0)
plt.show()
```