Data and Text Mining
MCDA5580

Master of Science in Computing and Data Analytics
Assignment-1 Report

Submitted by:
Jagwinder Kaur Dhillon (A00433543)
Rajalakshmi Ashokkumar (A00433403)
Gowtham Varma Kanumuru (A00433276)

SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

Table of Contents

Executive Summary

Sobeys Inc. is the second-largest food retailer in Canada, with over 1,500 stores operating across Canada. Sobeys wanted to analyze the data from its sales, to provide customers and also for different departments if it needs to invest in for more profits. As part of this, Sobeys has given the historical retail transaction data that was recorded in 2015 i.e. from 1st January 2015 to 14th September 2015 to a team of analysts to provide insights for its operation.

Objective

In order to understand the purchase behavior of customer and characteristics of products for the given Sobeys dataset, we have used k-means clustering. We use this method of unsupervised learning algorithm as we do not know the function of x and the output that it could produce (ie y).

Data Summary

The dataset from Sobeys has been loaded into MySQL database name dataset01.The tables that we are using to create our target tables is

• sales219: Containing the transaction data of its customer over several outlets.

Total No of Products: 32591
Total No of Customers:44469
Total Revenue: $13645221
Time span in Consideration: 814 days (2015-01-01 to 2005-09-14)

Product Analysis:

1. Data Model for Products:

From the “sales219” table the following columns have been extracted for product analysis.
We have considered the top 2000 products have been chosen based on the total revenue earned by each product.

Column Name	Description
ITEM_SK	Unique product serial number which uniquely identifies the product.
TOTAL_REVENUE	Total revenue generated from an individual item
BASKETS	No of distinct transaction of a particular product
DISTINCT_CUSTOMERS	No of distinct customers purchased a product
AVERAGE_PRICE	Average price of a product
BASKETS_BELOW_AVG_PRICE	Baskets below average price

Table 1-Columns for Products Table

2. Data Cleaning

While inspecting the data it was observed that there were negative values for the ITEM_QTY, therefore, a subset of dataset having only required columns with positive values for the ITEM_QTY is extracted as PRODUCTS table with an addition calculated column named unit price for each item purchased by dividing Selling_retail_amt with Item_qty.

An intermediate table Productsale is created from Products table with an additional column of Average Price. This calculated Average Price will be used further to count the number of times a product is purchased when its price is less than or equal to Average price.

Finally, data is grouped according to ITEM_SK in descending order for total revenue with selected features for clustering such as number of distinct transactions item appears in, number of distinct customer item purchased by, average selling price for the item, total revenue generated by the item, and number of transactions when unit price for item if less than or equal to average price. Only top 2000 items are considered for the clustering.

Additionally, ITEM_SK = 11740941, has been found as an outlier as the item was found to be banana which is a common item bought by everyone and does not contribute much in identifying the product sale pattern. Hence, it is removed from the data set and a cleaned data table has been prepared.

The extracted data has been cleaned to remove any outliers, errors and unwanted data.

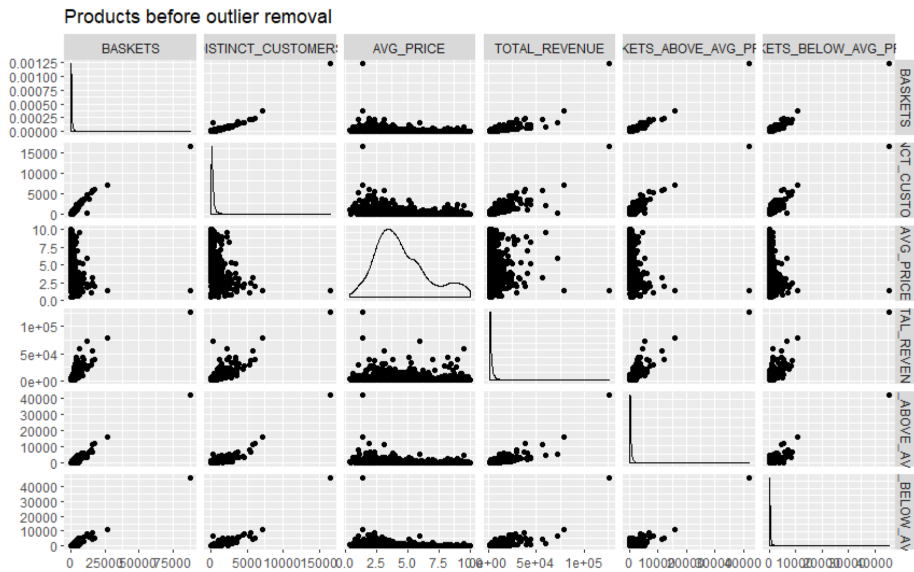

Figure 2-Products Before Outlier Removal

3. Data Profiling:

The attribute BASKETS_BELOW_AVG_PRICE, which indicates the products that are bought during discount sale. They belong to the monitory category of RFM.

4. Data Normalizing

After the data is removed with the outliers then we use the scaling function to scale the data and then use the data for the next process

5. Selecting Number of Clusters

After scaling, we need to identify the number of clusters, for that we use the withinSSRange function and plot the graph for range 1 to 50 and using the elbow method we identify the No of clusters

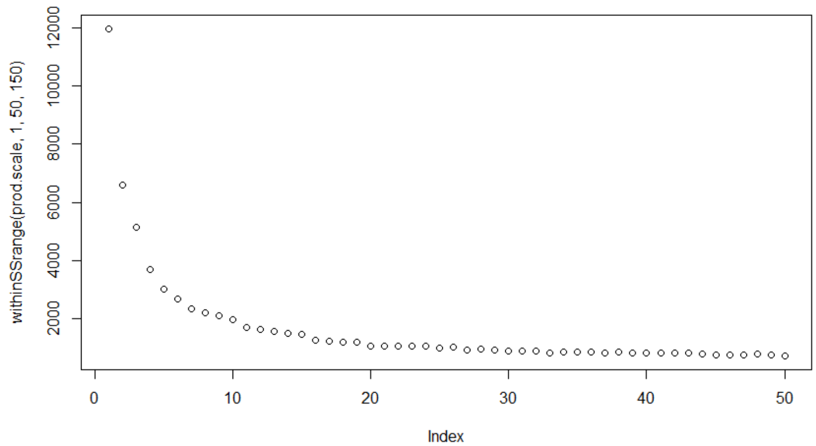

Figure 3-WithinSSRange Plot

From the above graph using Elbow Method we then find out that the optimum no of clusters is 5

6. Data De-normalizing and Cluster column addition

After number of clusters is decided. The data is again denormalized to get actual centroids of the clusters and cluster information is appended to the cleaned product table.

7. Data Analysis

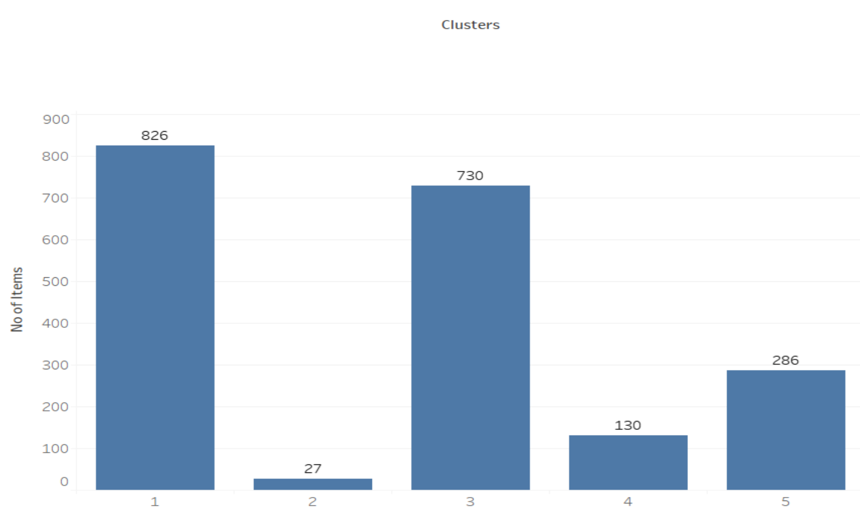

Figure 4-No.of Items Vs Clusters

According to the above figure cluster-1 also has highest number of customers.

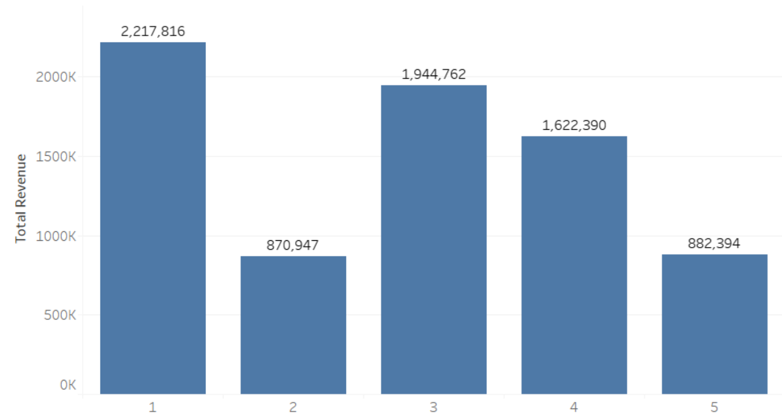

Figure 4-Total Revenue Vs Clusters

According to the above figure cluster-1 also has highest Total Revenue.

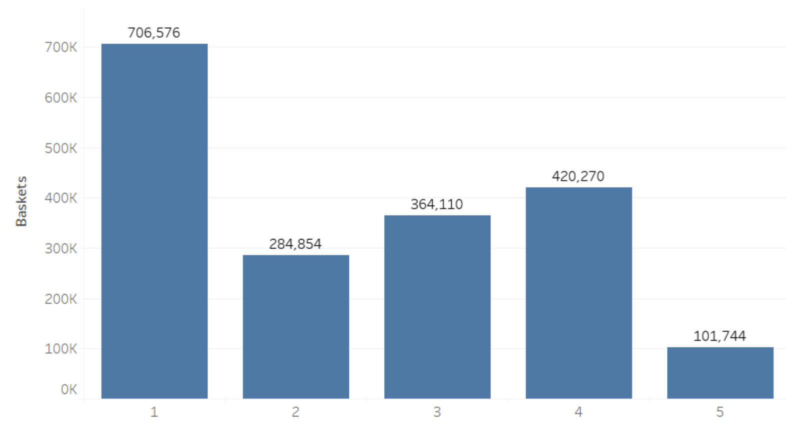

Figure 5-Sum Of Baskets Vs Clusters

According to the above figure cluster-1 has highest number of baskets.

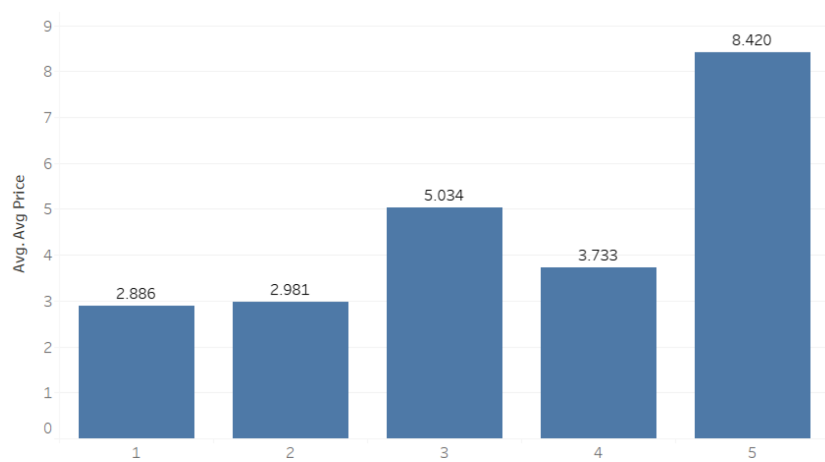

Figure 6-Avg of Average Price Vs Clusters

According to the above figure average product price is highest for cluster-5.

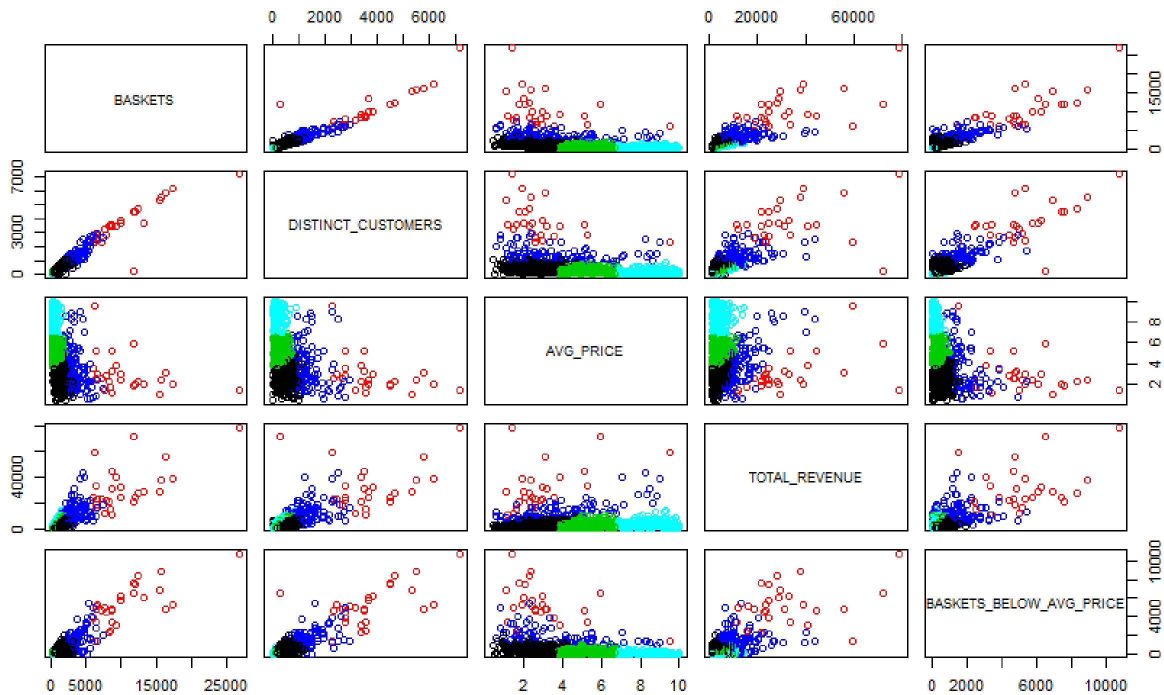

Figure 7-Clustering of All Attributes

8. Product Profiling

The following table shows description of different products and recommendations

Product Segment	Description	Recommendation
Cluster 1: Revenue Generators	• Highest revenue generating products (Dairy and meat are the main contributors) • Highest number of visits. • Highest count of distinct products • Buys products when they are on sale	• Products are already in demand hence maintaining enough stocks is strongly recommended.
Cluster 2: Heavy Runner	• Least revenue generator (Produce is the main contributor) • Third highest selling product • Buy repeated Products	• Deals on pricey items should be increased.
Cluster 3: Fast Movers	• Second highest revenue generator (Pops and sauces are the main sources) • Second highest selling product • Products having highest average price	• Fast moving products: Maintain sufficient stocks

	• Second highest count of distinct products	
Cluster 4: Attention Seeker	• Medium revenue generator • Second lowest customers • Lower count of distinct products sold • Second lowest average price	• Deals should be given on individual item sales to make customers prefer these products.
Cluster 5: Promotion required	• Second Lowest revenue generator • Buy products only when they are on sale • Products having highest average price of sale	• Products should be advertised and deals should be given. • Only optimum stock should be kept to avoid risk of loss in revenue.

Table 2-Segmentation for Products Table

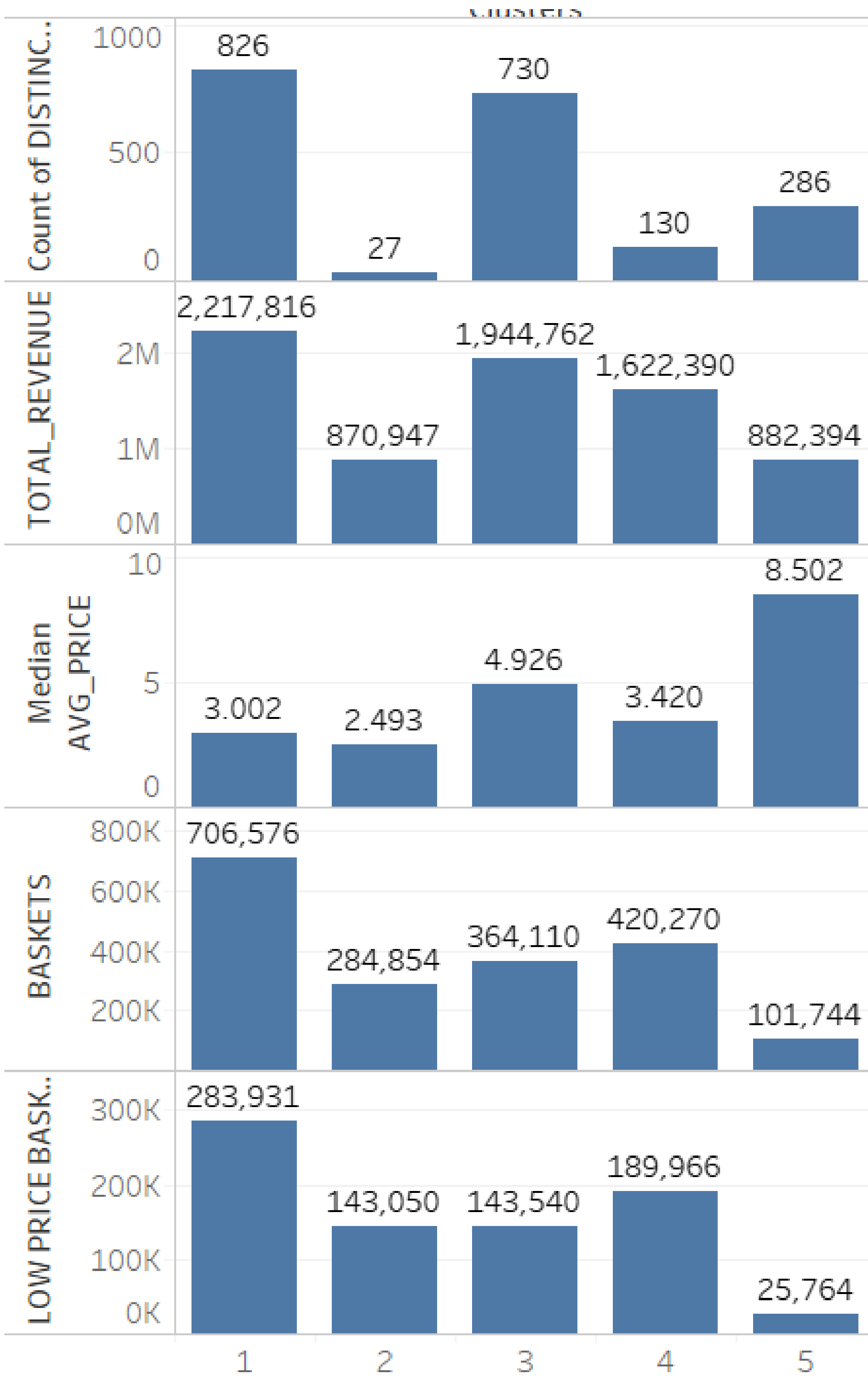

Figure 8-Comparison of All Attributes in each Clusters

Customer Analysis

The “sales219” dataset and RFM (recency, frequency and monetary) model have been used to analyze the customer behavior and performing the customer segmentation. In this report, the results are based on the top 2000 customers from the database sorted in descending order of the total amount of money spent of the customer in store in last 9 months data.

9. Data Preparation

The data has been extracted and manipulated in order to make meaningful data. The maximum date of the complete dataset is 2015-09-14, so “Days” is the Recency attribute of the RFM model. The frequency is represented by “Baskets” and monetary is by “Expenditure” column.

The following are the columns that have been selected and calculated from the dataset “sales219” for customer analysis and segmentation.

Column Name	Description
CUSTOMER_SK	Unique Customer id which identify each customer
ITEMS	Number of unique items bought by a specific customer
QUANTITY	Total quantity of items purchased by the customer
BASKETS	Number of times customer visit and make transaction
EXPENDITURE	Total amount of money spent by the customer
DAYS	Number of days ago was the customer's last purchase
AVG_BILL	Average amount of money spent per visit by the Customer
MORN_SHOP	Number of times the purchase was made in the morning
EVEN_SHOP	Number of times the purchase was made in the evening

Table 3-Columns for Customer Table

10. Data Cleaning

While inspecting the data it has been observed that data contain zero and negative values for the ITEM_QTY column which is ~ 58000 rows which may influence the clustering, hence, it is removed. The extracted data has been further cleaned to remove any outliers, errors and unwanted data.

In ggpairs plot, Customer_SK with value “1”is found out as outlier having very high value for all the columns and looks like “1” is default customer_sk assigned to all unknown customers. Therefore, removed as outlier and again ggpairs plot is ploted.

Again, in ggpairs plot a customer with Customer_sk = “64593270” has been identified as outlier after initial cleaning. This customer is frequent visitor with very high total expenditure but low average bill. It can be inferred from the attribute values that this customer did most of the shopping from Sobeys therefore does not contribute much towards finding customer behavior. Figures 9, 10 and 11 show the ggpairs plots to identify outliers.

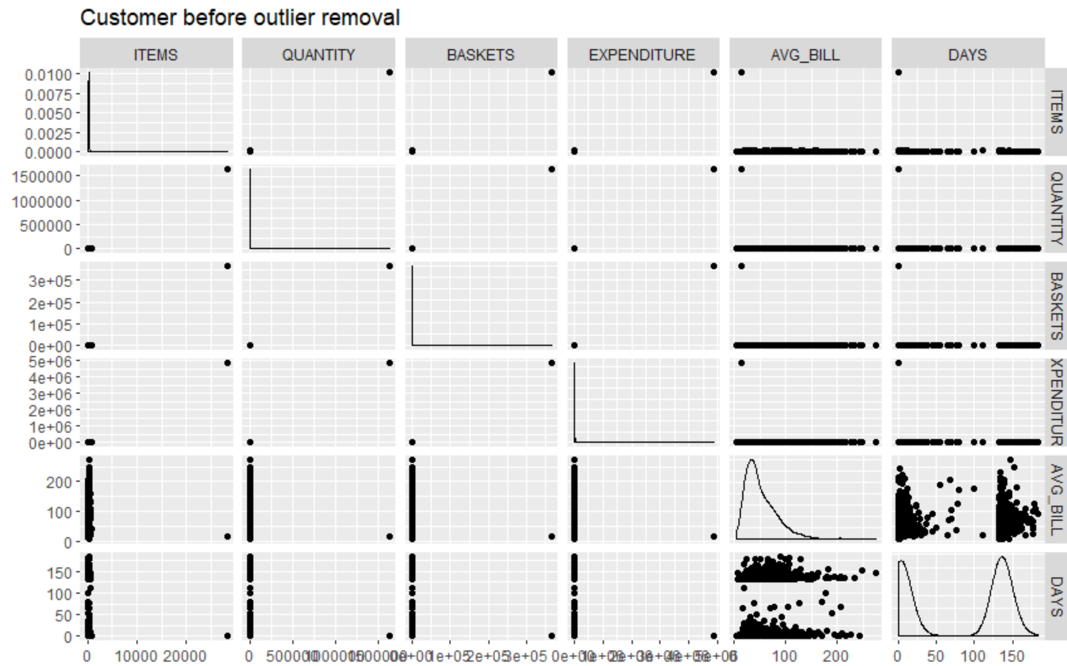

Figure 9-Graph Before Removal of Outliers

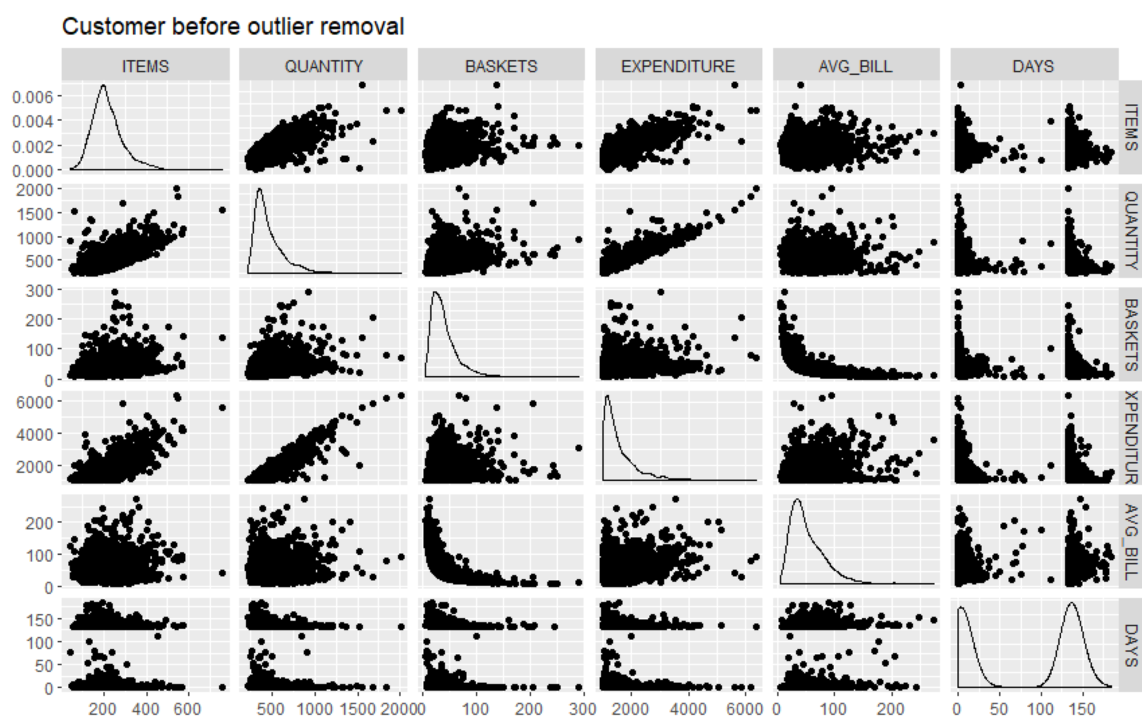

Figure 10-Graph After Removal of 1 Outliers

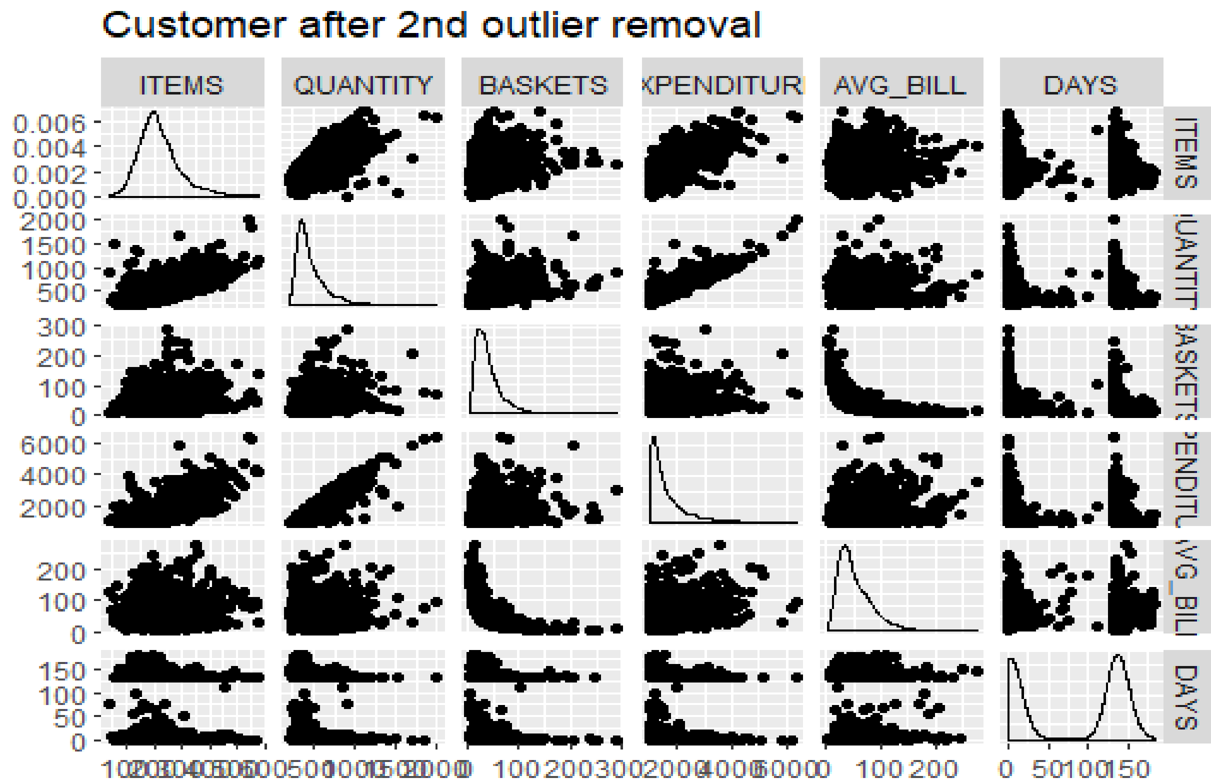

Figure 11-Customer after 2nd outlier removal

11. Data Normalizing

After removing outliers, data scaling was performed to normalize the data. This is an important step because each column having different scale need to be on same scale to perform clustering.

12. Selecting Number of Clusters

After scaling, we need to identify the number of clusters. This is done by defining a function named “withinSSRange”. This function takes the range of numbers and number of iteration as argument to plot the graph for the range based on tot.withinss calculation. This graph as shown in figure 12 is known as elbow method that helps in deciding the value for K i.e number of customers.

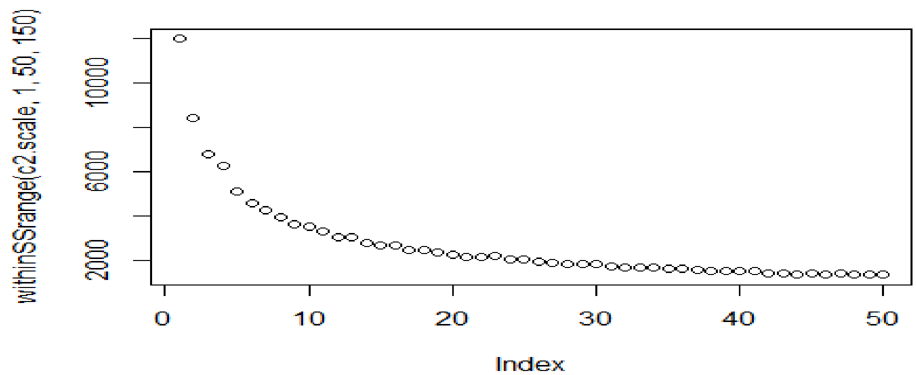

Figure 12- Elbow Plot

From the above graph using Elbow Method we then find out that the optimum no of clusters is 6.

13. Data De-normalizing and Clustering

After number of clusters is decided. The data is again denormalized to get actual centroids of the clusters and cluster information is appended to the cleaned product table.

14. Data Analysis

Kmeans clustering is used to perform the clustering. Clustering has been done on five features namely Items, Quantity, Baskets, Expenditure, Average Bill and Days. Figure 13 shows the results of clustering.

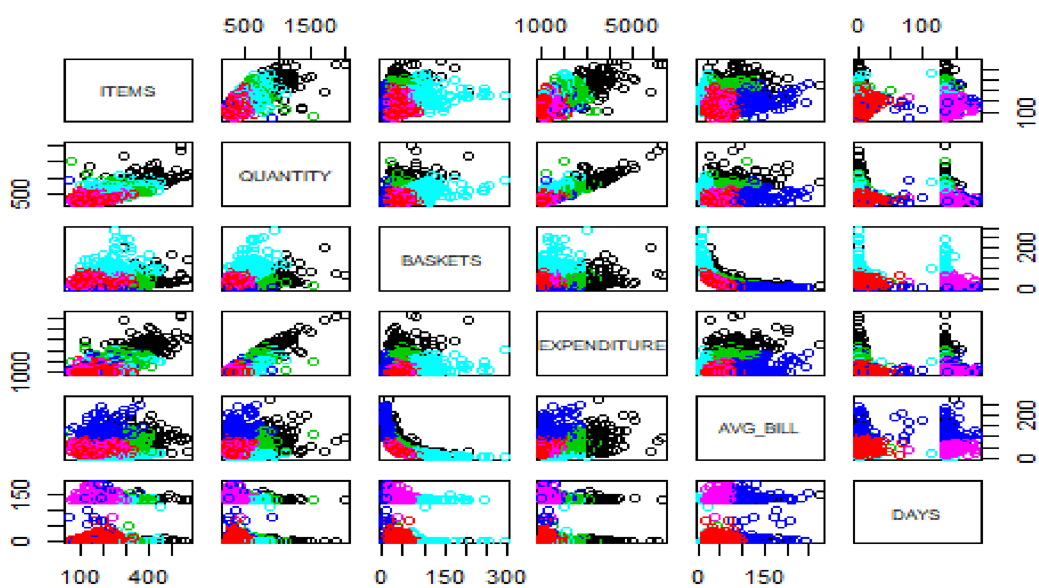

Figure 13- Kmeans clustering Plot

15. Customer Profiling

Customer visited store on last 30 days is considered active.

Customer Segment	Description	Recommendation
Cluster 1: Frequent Buyers	• 6% of total customers • Average number of visits per customer is high • 50% active customer • Average bill is medium • Contribute 12% towards total expenditure • Highest average expenditure	Keep day to day products such as in stock to keep them visiting again
Cluster 2: Champions	• 30% of total customer • Highest number of distinct items • 99% active customer • High Average bill • Contribute 24% towards total expenditure • Medium number of visits per customer • Medium average expenditure	Already actively buying products.
Cluster 3: Potential customers	• 14% of total customer • Medium number of visits per customer • High average expenditure • High Average bill • 50% active customer • 18% of total expenditure	As average bill is high so can be encouraged to come frequently by giving gift voucher
Cluster 4: Impulse shoppers	• 10% of total customer • Low average number of visits • 10% of total expenditure • Very high average bill • 25% active customer	Give them points card to increase there visit and spending
Cluster 5: Bargain hunter	• 9% of total customer • 9% of total expenditure • Very high average number of visits • 75% active customer	Sales and discounts on high price items

	• Medium Average expenditure • Low Average Bill	
Cluster 6: Attention seeker	• 31% of total customer • 24% of total expenditure • Medium average number of visits • Zero Active customer • Medium average bill • Very high distinct number of items	Promote online shopping with discount coupon

Table 4-Segmentation for Customer Table

	# of CUSTOMERS	DISTINCT ITEMS	TOTAL QUANTITY	# OF TRANSACTIONS	TOTAL EXPENDITURE	AVERAGE BILL	ACTIVE CUSTOMER	AVERAGE EXPENDITURE	Average Visits	% of total Expenditure	% of Total Customer
1	115	43947	111995	6354	400860.106	44.30175	67	3485.740052	55.25217	12.70470575	5.755755756
2	594	111384	222739	20468	762245.03	61.53853	587	1283.240791	34.45791	24.1583003	29.72972973
3	275	78425	174060	10969	595213.2287	61.53853	129	2164.411741	39.88727	18.86445874	13.76376376
4	208	41861	84157	2832	304643.6003	113.6396	40	1464.632694	13.61538	9.655256893	10.41041041
5	186	46170	106090	18889	308762.3557	18.05247	127	1660.012665	101.5538	9.78579514	9.309309309
6	620	113757	229364	21353	783485.3612	42.24415	0	1263.686067	34.44032	24.83148317	31.03103103

Table 5-Analysis based on clustering

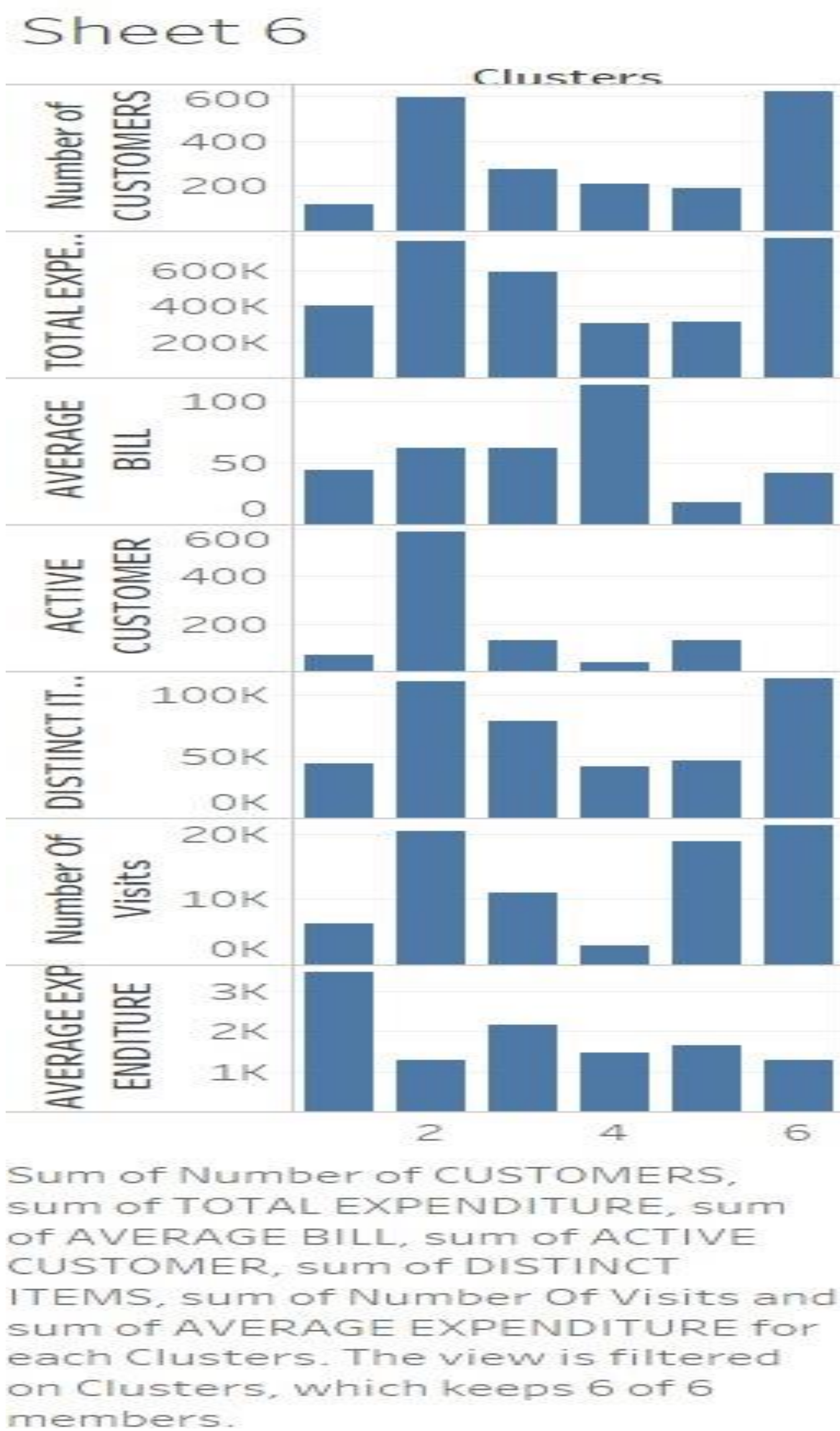

Figure 5-comparing attributes for each cluster

16. Additional Analysis

Common products across all top 5 customers in each cluster.

Cluster 1	Red Cluste
	Comp Eggs
	Bananas
	Green Seed
	Lemons Lar
	English Cu
	Broccoli
Cluster 2	Red Cluste
	Bananas
	Hothouse
Cluster 4	Bananas
	Extra Lean
Cluster 5	Bananas
Cluster 6	Bananas

Top 5 revenue generating Products in each Cluster

Clusters	**top 5 Products**
Cluster 1	Windsor sa
	Crispy Chi
	Scotsbrn T
	Comp Eggs
	Pork Belly
Cluster 2	Avacado Ha
	On Line Lo
	GourmMin B
	Comp Eggs
	Red Seedle
Cluster 3	Coca Cola C
	Bagel Mont
	New World S
	Comp Extra
	C/Farm Egg
Cluster 4	Chicken Br
	Unknown Pr
	Lean Groun

	Sensatns C
	Haddock Fi
Cluster 5	Extra Lean
	Celebration
	Fresh Atl
	BlueGse Bn
	NewWorld D

Appendix

1. **Product Cluster SQL Code**

1.a)

```
CREATE TABLE products AS
SELECT ITEM_SK, CUSTOMER_SK, TRANSACTION_RK, ITEM_QTY, SELLING_RETAIL_AMT,
SELLING_RETAIL_AMT.ITEM_QTY as UNIT_PRICE TIME
from dataset01.sales219
WHERE ITEM_QTY >0;
```

1.b)

```
CREATE TABLE productsale AS
select *
from
(
(SELECT * FROM jk_dhillon.products)  aa
natural join
(SELECT ITEM_SK, AVG(SELLING_RETAIL_AMT/NULLIF(ITEM_QTY,0)) FROM jk_dhillon.products
GROUP BY ITEM_SK) AVERAGE_PRICE
)
```

1.c)

```
CREATE table P_CLUSTERS AS
SELECT
ITEM_SK,
count(distinct TRANSACTION_RK) as BASKETS,
count(distinct CUSTOMER_SK) as DISTINCT_CUSTOMERS,
avg(SELLING_RETAIL_AMT/NULLIF(ITEM_QTY,0)) as AVG_PRICE,
sum(SELLING_RETAIL_AMT) as TOTAL_REVENUE,
COUNT(DISTINCT (CASE WHEN UNIT_PRICE > AVERAGE_PRICE THEN TRANSACTION_RK END)) as
BASKETS_ABOVE_AVG_PRICE,
COUNT(DISTINCT(CASE WHEN UNIT_PRICE <= AVERAGE_PRICE THEN TRANSACTION_RK END))
as BASKETS_BELOW_AVG_PRICE,
SUM(CASE WHEN UNIT_PRICE > AVERAGE_PRICE THEN SELLING_RETAIL_AMT END) as
REVENUE_ABOVE_AVG_PRICE,
SUM(CASE WHEN UNIT_PRICE <= AVERAGE_PRICE THEN SELLING_RETAIL_AMT END) as
REVENUE_BELOW_AVG_PRICE
FROM jk_dhillon.productsales
GROUP BY ITEM_SK
ORDER BY TOTAL_REVENUE DESC
LIMIT 2000;
```

2. Product Cluster R Code

```
library(ggplot2)
library(GGally)
library(DMwR)

P_CLUSTERS <- read.csv("C:/Users/Jagwinder Dhillon/Desktop/P_CLUSTERS.csv",
stringsAsFactors=FALSE)

View(P_CLUSTERS)

prod<- P_CLUSTERS[, c(1:5,7)]

View(prod)
ggpairs(prod[, which(names(prod) != "ITEM_SK")], upper = list(continuous = ggally_points),lower
= list(continuous = "points"), title = "Products before outlier removal")

prod.clean <- prod[prod$ITEM_SK != 11740941, ]

prod.scale = scale(prod.clean[-1])

withinSSrange <- function(data,low,high,maxIter)
{
  withinss = array(0, dim=c(high-low+1));
  for(i in low:high)
  {
    withinss[i-low+1] <- kmeans(data, i, maxIter)$tot.withinss
  }
  withinss
}

plot(withinSSrange(prod.scale,1,50,150))

pkm = kmeans(prod.scale, 5, 150)

prod.realCenters = unscale(pkm$centers, prod.scale)

clusteredProd = cbind(prod.clean, pkm$cluster)

plot(clusteredProd[,2:6], col=pkm$cluster)
```

```
write.cv(clusterProd,"ProductCluster.csv")
```

3. **Customer Cluster SQL Code**

```
CREATE TABLE C_CLUSTERS AS
SELECT
CUSTOMER_SK,
COUNT(DISTINCT ITEM_SK) as ITEMS,
SUM(Item_qty) as QUANTITY,
COUNT(DISTINCT TRANSACTION_RK) as BASKETS,
SUM(SELLING_RETAIL_AMT) as EXPENDITURE,
SUM(SELLING_RETAIL_AMT)/COUNT(DISTINCT TRANSACTION_RK) as AVG_BILL,
MAX(date) as RECENT_DATE,
DATEDIFF('2015-09-14', MAX(date)) as DAYS,
COUNT(DISTINCT(CASE WHEN time <= '15:00:00' THEN TRANSACTION_RK END)) as
MORN_SHOP,
COUNT(DISTINCT(CASE WHEN time > '15:00:00' THEN TRANSACTION_RK END)) as EVEN_SHOP
FROM dataset01.sales219
GROUP BY CUSTOMER_SK
ORDER BY EXPENDITURE DESC
LIMIT 2000
```

4. **Customer Cluster R Code**

```
library(ggplot2)
library(GGally)
library(DMwR)

C_CLUSTERS <- read.csv("C:/Users/Jagwinder Dhillon/Desktop/C_CLUSTERS.csv",
stringsAsFactors=FALSE)

View(C_CLUSTERS)

C <- C_CLUSTERS[c(1:6,8)]

ggpairs(C[, which(names(C) != "CUSTOMER_SK")], upper = list(continuous = ggally_points),lower
= list(continuous = "points"), title = "Products before outlier removal")

C.clean <- C[C$CUSTOMER_SK != 1, ]

ggpairs(C[, which(names(C) != "CUSTOMER_SK")], upper = list(continuous = ggally_points),lower
= list(continuous = "points"), title = "Products before outlier removal")
```

```
C.clean <- C[C$customer_sk!= 64593270, ]

C.scale = scale(C.clean[-1])

withinSSrange <- function(data,low,high,maxIter)
{
  withinss = array(0, dim=c(high-low+1));
  for(i in low:high)
  {
    withinss[i-low+1] <- kmeans(data, i, maxIter)$tot.withinss
  }
  withinss
}

plot(withinSSrange(C.scale,1,50,150))

ckm = kmeans(C.scale, 6, 150)

C.realCenters = unscale(ckm$centers, C.scale)

clusteredcust = cbind(C.clean, ckm$cluster)

plot(clusteredcust[,2:7], col=ckm$cluster)

write.cv(clustercust,"daysK6.csv")
```

References

1)Class Materials and Tutorials Provided by Professor’s and Tutors