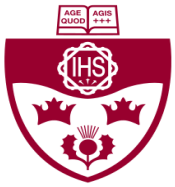


SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

Master of Science in Computing and Data Analytics

Data and Text Mining

MCDA 5580

Assignment 1

Submitted To:

Ms. Trishla Shah

Submitted By:

Haozhou Wang (A00431268)

Hemanchal Joshi (A00433394)

Rishi Karki (A00432524)

TABLE OF CONTENTS

1. Executive Summary

The report is prepared on the ground of collection of the transaction of online retail store with the primarily aim to perform clustering analysis using K-means algorithm that would lead to effective number of clusters of product and customer. In addition, with a resultant clusters of product and customer, the main theme was to aid the Online Retail store in identifying the typical patterns so that this business house can target the potential audience of customers with respect to their underlying nature of products. Similarly, as of this business can gain abundance of profit and retain the customer base simultaneously with the improvement of quality of service through timely knowledge based decision making. Here, we took a sample data of top 2,000 rows from the OnlineRetail dataset on the basis of the revenue arranged in an descending order to focus on the various attributes of the products as well as the customers.

Similarly, profiling for Customer and Product based on as determined optimal cluster size is done in order to analyze and derive the customer's buying traits and product differentiation which could help the business house to formulate the various marketing strategy so as to have comparative and competitive advantage among other prevailing retail stores. Based on the clustering analysis, we have come up with some recommendation for each of the Customer and Product cluster with the help of the k-means[1] algorithm which is an unsupervised algorithm.

2. Objective

The main objective of the analysis of this dataset is to recognize the various pattern of purchase and sales in the particular retail store and plan the future sales according to the results obtained. The detailed purpose of conducting this analysis is

- To point out and emphasize on the attribute that is generating most of the revenue for the organization.
- Profile the two objects product and customer and make various inferences regarding their attribute and their behaviour.
- To categorize customers and products into different behavioural group analyzing their previous data.

3. Data Summary

The given "OnlineRetail" table has total record of 53,761 rows where attributes include of InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country and InvoiceDateTime. While observing this bulk of data, it does contain lots of noisy and inconsistent data. Thus, prior to analysis in our preliminary steps, Data Cleaning should be done to get rid of those anomalies. For this, we wrote a SQL query to extract the basic attributes of the top 2000 data that gave the highest revenue and categorize them into two different tables **ProductCluster table** and **CustomerCluster table** using the where clause which cleaned almost all the negative and null valued data

WHERE CustomerID!="0" AND InvoiceNo!="0" AND UnitPrice!=0

Here, we removed the records with InvoiceNo, CustomerID and UnitPrice having negative and null values and form the new intermediary table as "Online Retail". For the purpose of cluster analysis, only top 2000 records for either of Product and Customer were taken into consideration as of each respective cluster table.

We exported the table as a CSV file to import it into R and summarize the available resultant data.

- **ProductCluster Summary**

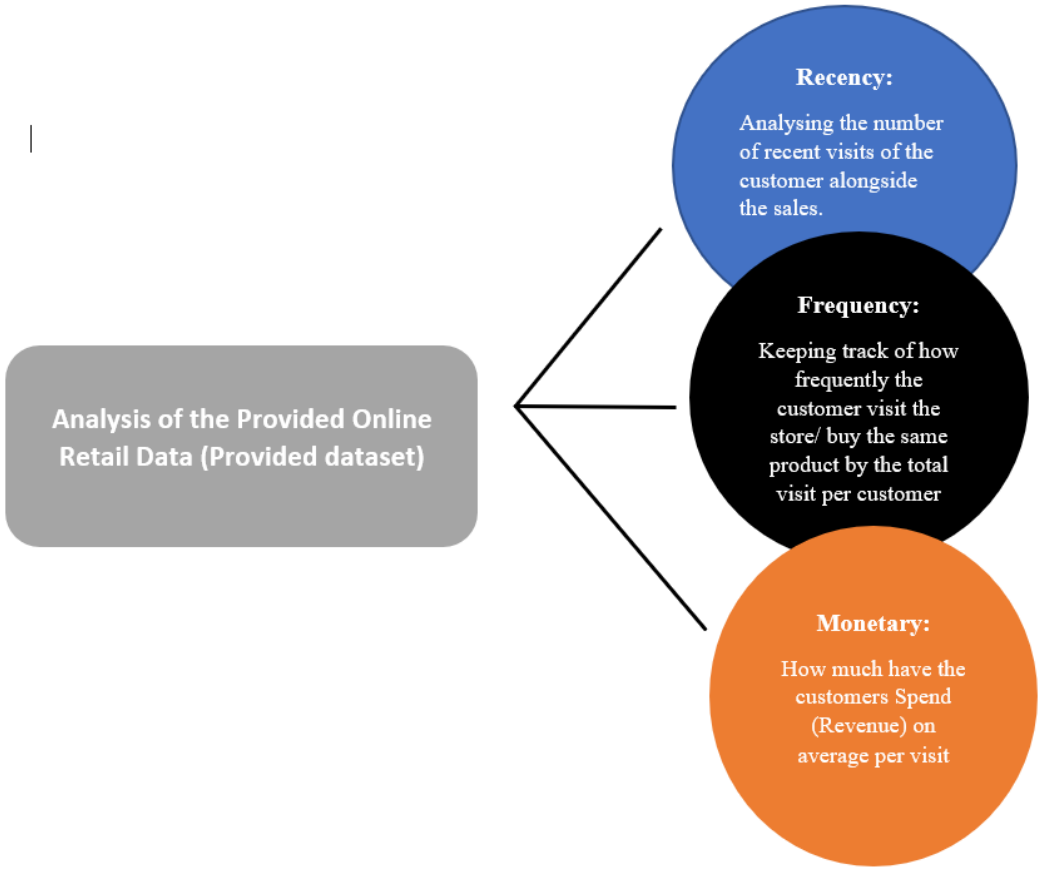
```
> summary(ProductCluster[-1])
     Visits           Sales         Total_Revenue      Distinct_Buyers  Sales_Frequency
 Min.   :   1.0   Min.   :   51.0   Min.   :   503.8   Min.   :  1.00   Min.   :    1.00
 1st Qu.:  69.0   1st Qu.:  517.8   1st Qu.:   936.2   1st Qu.: 55.75   1st Qu.:    5.88
 Median : 121.0   Median : 1219.0   Median :  1857.1   Median : 91.00   Median :    9.83
 Mean   : 180.3   Mean   : 2437.8   Mean   :  4186.9   Mean   :121.75   Mean   :   54.47
 3rd Qu.: 227.0   3rd Qu.: 2605.5   3rd Qu.:  4336.6   3rd Qu.:158.00   3rd Qu.:   15.55
 Max.   :1977.0   Max.   :80995.0   Max.   :168469.6   Max.   :881.00   Max.   :80995.00
```

- **CustomerCluster Summary**

```
> summary(CustomerCluster[-1])
 Total_Products     Distinct_Products Total_Revenue         Visits          Loyalty
 Min.   :    80.0   Min.   :   1.0    Min.   :   742.3   Min.   : 1.000   Min.   :   29.68
 1st Qu.:   633.8   1st Qu.:  44.0    1st Qu.:  1100.0   1st Qu.: 3.000   1st Qu.:  273.02
 Median :  1057.5   Median :  74.0    Median :  1769.0   Median : 5.000   Median :  387.96
 Mean   :  2330.2   Mean   : 102.6    Mean   :  3917.0   Mean   : 7.334   Mean   :  608.11
 3rd Qu.:  1938.8   3rd Qu.: 126.0    3rd Qu.:  3173.4   3rd Qu.: 8.000   3rd Qu.:  566.43
 Max.   :196915.0   Max.   :1787.0    Max.   :277335.8   Max.   :209.000   Max.   :84236.25
```

4. Design Approach

Basically, clustering and analysis of a financial institution which deals with sales and customer are approached via RFM (Recency, Frequency, Monetary) model for further analysis.



We worked basically on the Frequency and Monetary part of the RFM approach which led us to the resultant data and conclusion. As we didn't take the time and date parameter in to consideration working on recency was not an option.

For the Monetary evaluation revenue was calculated which also calculated the

Based on OnlineRetail dataset, clustering analysis for the Customer and Product was performed according to following approach:

a. First of all, OnlineRetail dataset was used a source point of data.
b. Along the provided feature, additional feature was selected, engineered and queried in creation for either of Product and Customer cluster table from the dataset to support the analysis.
c. For the analysis and simplicity, the clustered table was shrinked down as only top 2000 records for customers and products was taken into consideration based on revenue generated.
d. The data generated from dataset was cleaned and outliers were removed.
e. Normalization was done based on scaling function.
f. Appropriate number of clusters determined based on elbow plot diagram and then clustering operation was done.
g. Denormalization of the data was done reversing the scale function
h. Lastly, metadata from original dataset i.e "OnlineRetail" was used to create customer and product profile based on clustering results

5. Feature/Selection

Based on the requirements, we need to select features from products and customers. Initially we selected the features mentioned in the document as follows:

- Number of distinct customers who buy the product (COUNT(distinct CustomerID))
- Revenues (SUM(Quantity*UnitPrice))
- Number of visits in which the product is bought (COUNT(distinct InvoiceNo)

And as for the additional attribute, **initially we planned to use the country (I n which country which product is sold the most).**

But after the through discussion, **we concluded that this feature cannot be used as most of products are heavily sold in United Kingdom. We found that out of 2,000 sample data that generated the top revenue out of the dataset, 1998 products were that sold in United Kingdom, so Country cannot be taken an attribute in such the small sorted chunk of data.** Then we decide to choose other features as an additional attribute. So, the final attributes for both the tables can be summarized as follows:

- **Sales_Frequency** (SUM(Quantity)/Count(DISTINCT InvoiceNo) **e.g. The product having the highest frequency is getting sold faster in few visits**) or total products sold by total visit. **(for Product Cluster)**
- Number of products bought (SUM(Quantity))
- Number of distinct products bought (COUNT(DISTINCT StockCode))
- Revenues (SUM(Quantity*UnitPrice))
- Number of visits(COUNT(DISTINCT InvoiceNo))
- **Loyalty** SUM(Quantity*Unit Price)/COUNT(DISTINCT InvoiceNo)) which gives the Total money spend by total visit **(for customer cluster; e.g. the customer having highest loyalty is the most loyal customer)**

  Hence, we have selected enough features from the database and can convert them to our train dataset.

6. Data cleaning / Outliers removal

In a particular dataset, there is always the presence of some inconsistent data which act as an outlier in the analysis process. These kinds of data are not particularly anomalies but do not follow a similar pattern to the remaining data or lie beyond a certain limit of the designated data. The anomalies or the errors in the dataset have already been dealt with while creating the CustomerCluster and ProductCluster table.

To remove such outliers there are various methodologies including the manual and coded ones. We had various group discussions and took help and ideas with other groups in order to implement the proper Outlier removal technique. We used ggplot library in R to draw a scatter plot and locate all the outliers for both CustomerCluster and ProductsCluster.

6.1. Inconsistent data with Outliers

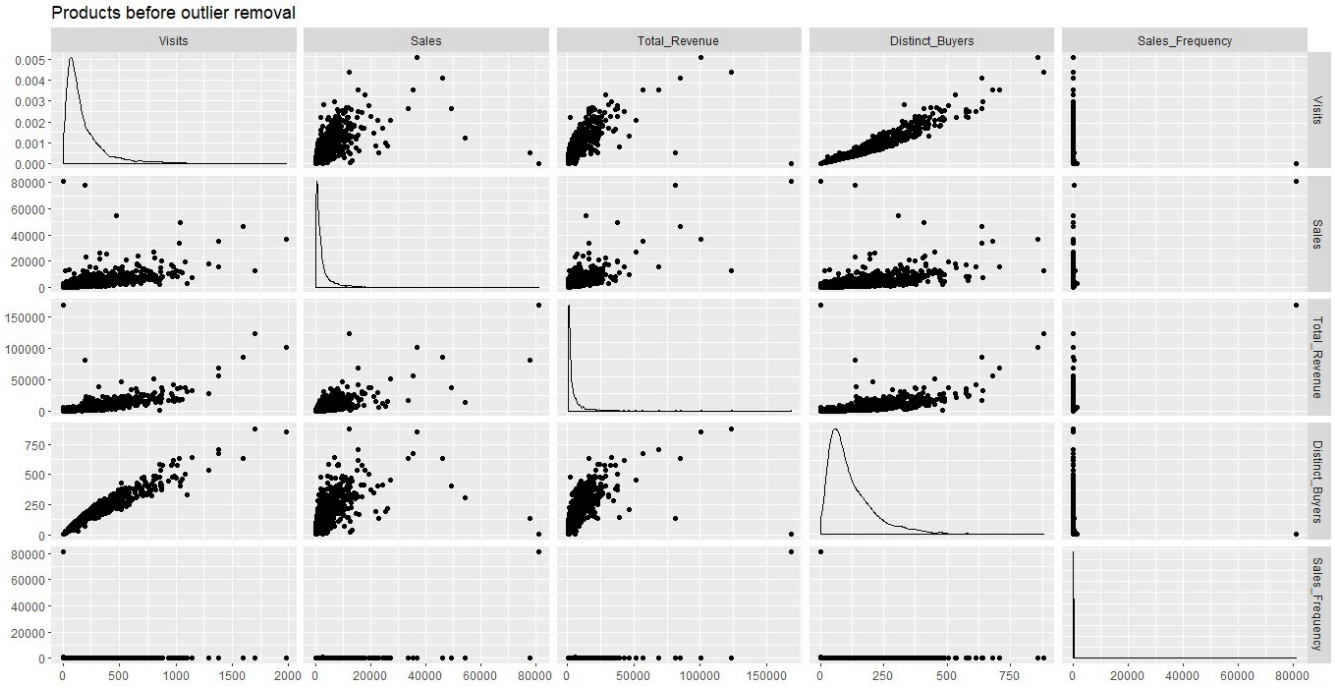


Fig: ProductsCluster data before outlier removal

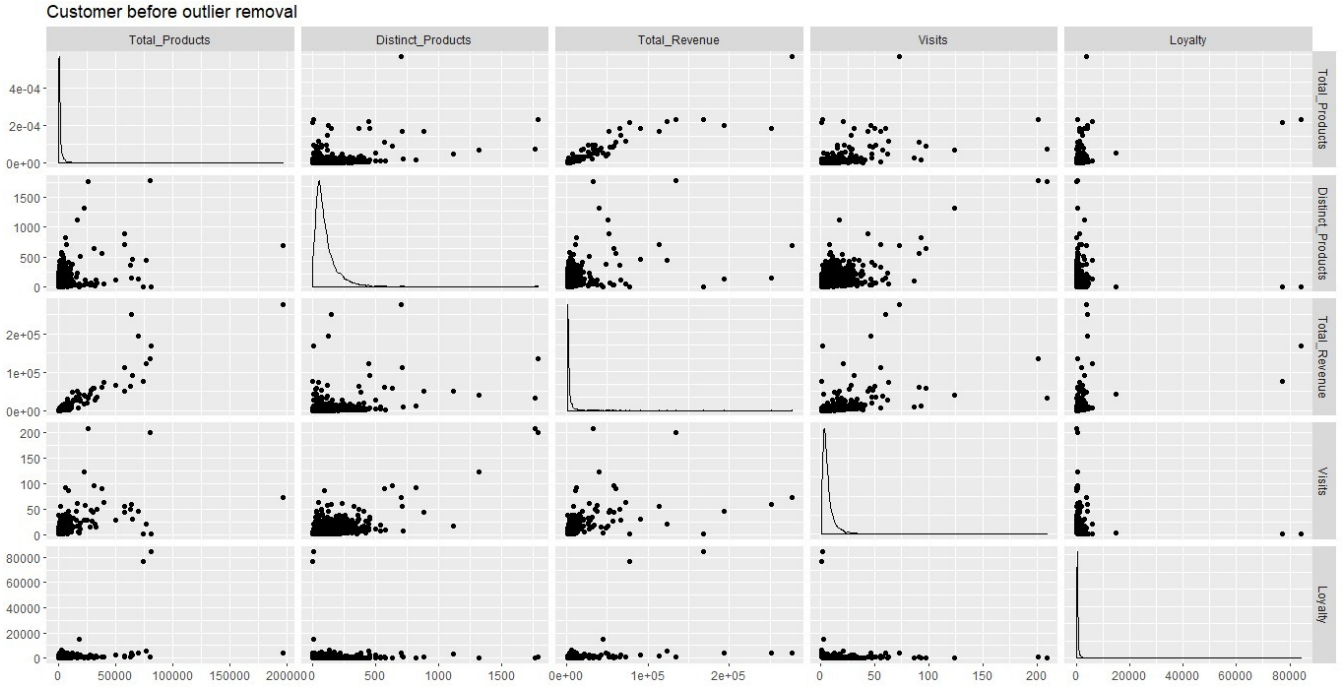


Fig: CustomersCluster data before outlier removal

Some of the approaches we tried to implement are:

6.1.1. Using Scatterplot (ggplot2)

We used ggplot2 package in R in order to draw a scatter plot graph with respect to all corresponding attributes. For this, we used two methods:

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

ggpairs(Customer[, which(names(Customer) != "CustomerID")], upper = list(continuous = ggally_points),lower = list(continuous = "points"), title = "Customers before outlier removal")

- **Graph analysis and Manual analysis of Outlier rows**

  Firstly, we used the graph to analyze the outliers that lie beyond the range that rest of the data follows. Then, we picked up the row number manually from each of the columns which have an inconsistent value i.e. very high or very low and then writing the following code in R and pointing out all the rows that needs to be omitted. This was a lengthy and tiring process and did not reduce all the outliers.

  Product.clean <- Product[Product$Stockcode != 14646,]

  Customer.clean <- Customer[Customer$CustomerID!=12748,]

- **Graph analysis and Threshold allocation**

  Secondly, we tried focusing on the scatterplot and looked for the distance between the clustered and the scattered value. Then, we set a threshold above which all the rows containing those maximum values will be omitted.

  Product.clean <- Product.clean [-which (Product.clean$Sales >40000),]

  Customer.clean <- Customer.clean[-which(Customer.clean$Visits >40),]

6.1.2. Using Boxplot

We used boxplot in order to identify the maximum value that lies far from the aligned value for a particular attribute and then performed following one after another.

boxplot(Customer$Total_Revenue)

boxplot(Product$Total_Revenue)

- **Storing outliers into a vector (Quick and non-standard method)**

  First we tried the heat and trial method. This is not a standard format for outlier removal but as we found it after a thorough search and study, we gave it a try but the result was not optimal because it removes a lot of rows which makes the dataset very small and not convincing enough.
  Outliers <-boxplot(CustomerCluster$Visits, plot=FALSE)$out

- **Use of Quantile function and High-Threshold allocation**

  Lastly, after going through a lot of difficulties, we got this idea from various sources when we looked for help. We considered the box plot into three quartiles and calculated the i**nterquartile range**. After that, we calculated the maximum and minimum range for the data to exist from the range. Finally we ignored the data that was less than the maximum and got out clean data shortlisted for final analysis.

  Q1 <- quantile(Product$Total_Revenue, probs = 0.25)

  Q3 <- quantile(Product$Total_Revenue, probs = 0.75)

  IQR <- Q3-Q1

  minimum <- Q1-3*IQR

  maximum <- Q3+3*IQR

  Product.clean <- Product[-

  which(Product$Total_Revenue>maximum),]

6.2. Clean Data After Outlier Removal

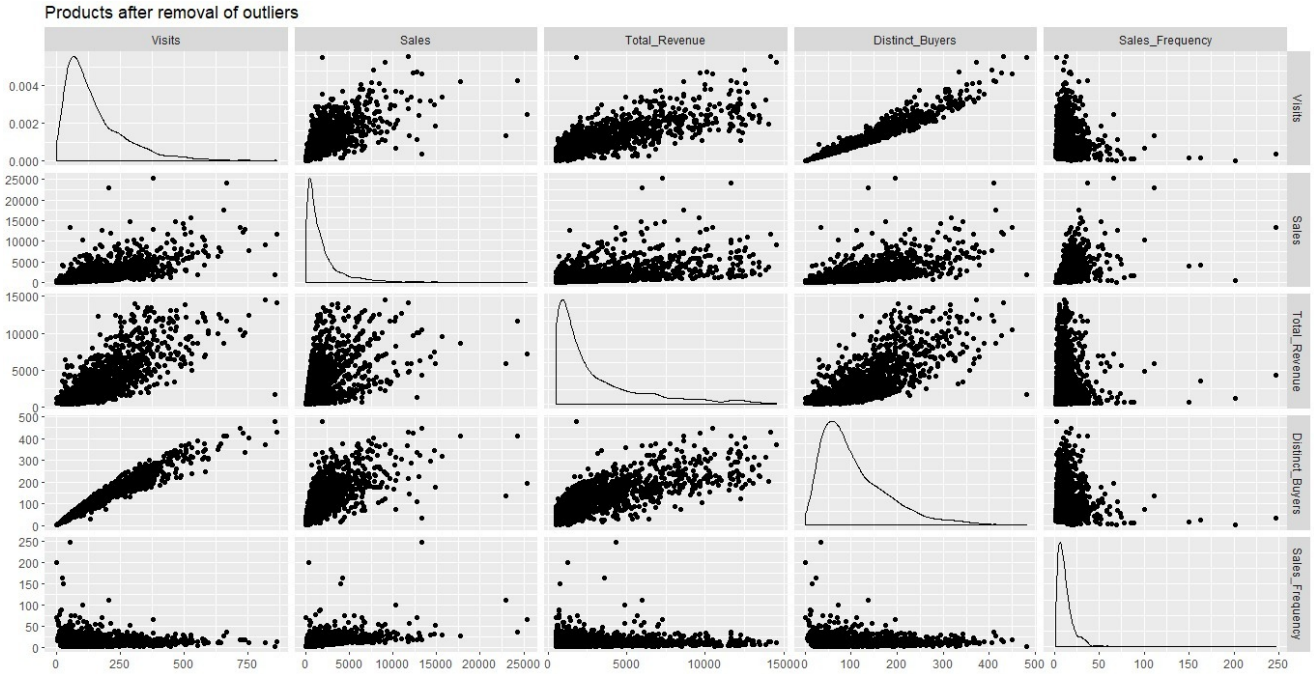


Fig: ProductsCluster data after outlier removal

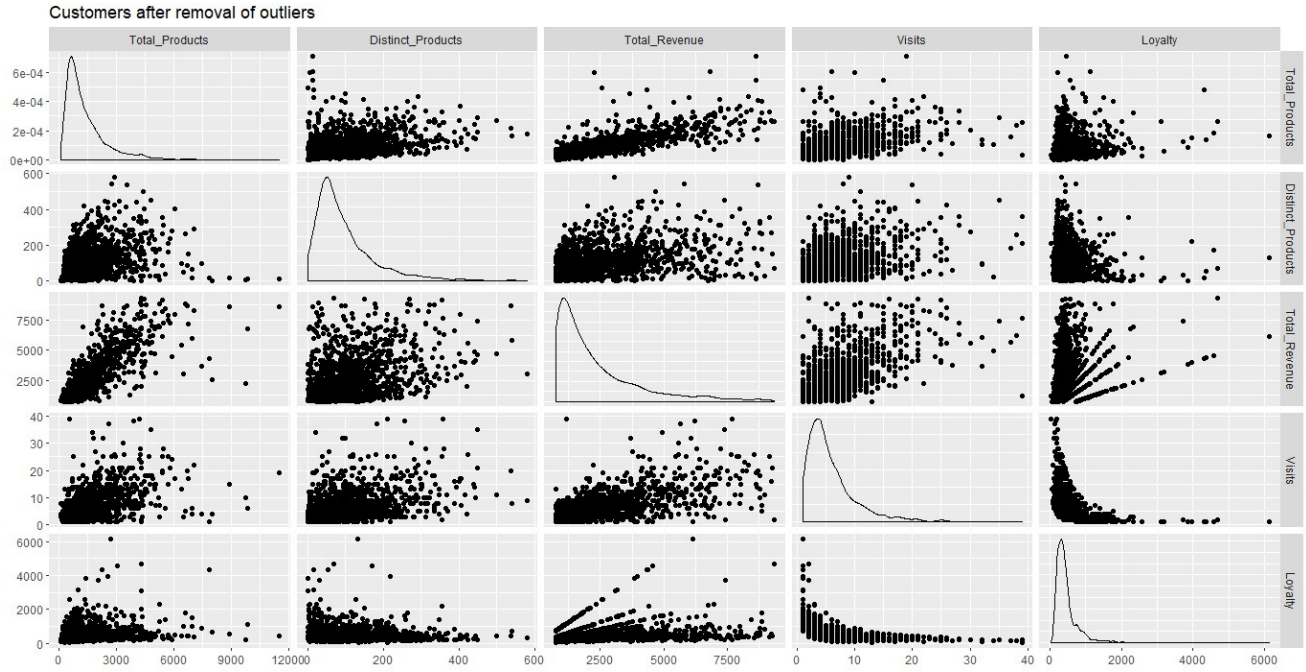


Fig: CustomersCluster data after outlier removal

In the end we normalize the available data in order to categorize all data into similar range.

7. Cluster Analysis

After cleaning the data and arranging it in order to perform the analysis operations on it, we performed the k-means clustering algorithms in order to cluster the dataset into various group. To perform the k-means algorithm we need to be decisive of the number of cluster we want to form the graph plot which forms a elbow kind of structure.

In elbow plot method we use the **sum of squared Errors (SSE). Elbow method works well only when the dataset can be well clustered** whereas it fails to work properly if clusters cannot formed well. **If the data is not very clustered than we can use other approaches like silhouette clustering which is a measure if how similar one's own cluster is with the other cluster.**

After getting the elbow point which means the number of clusters you should make while using the kmeans function in R, we apply the k-means function. Then, We calculate the centroid of each cluster, tot.withinss and then analyze it accordingly. Similar resultant procedures are followed for both the dataset.

7.1. Product Clustering

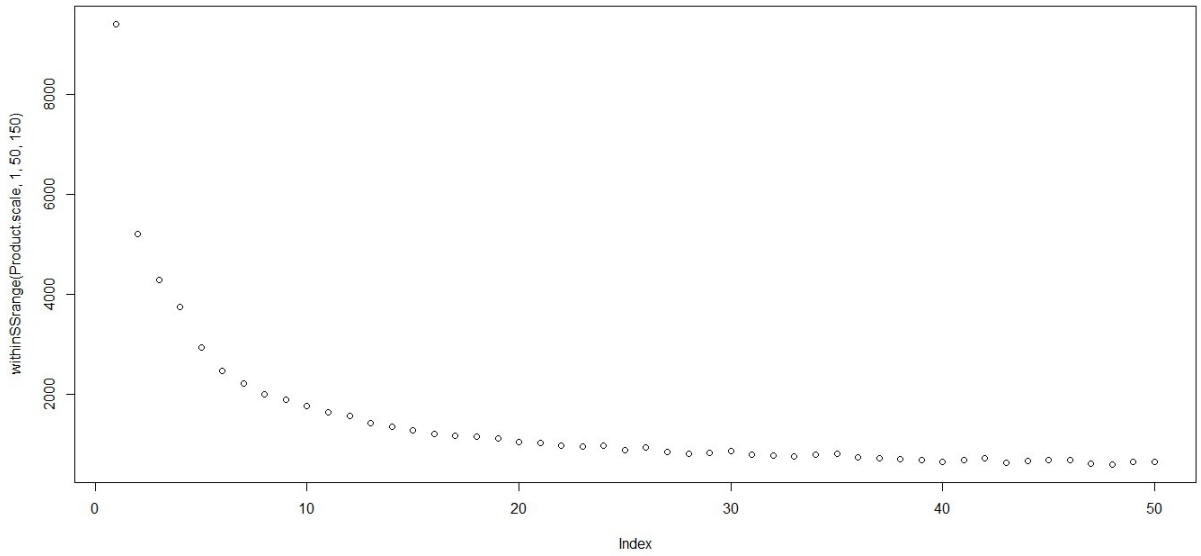


Fig: Elbow point for Product dataset denotes that there should be 5 clusters

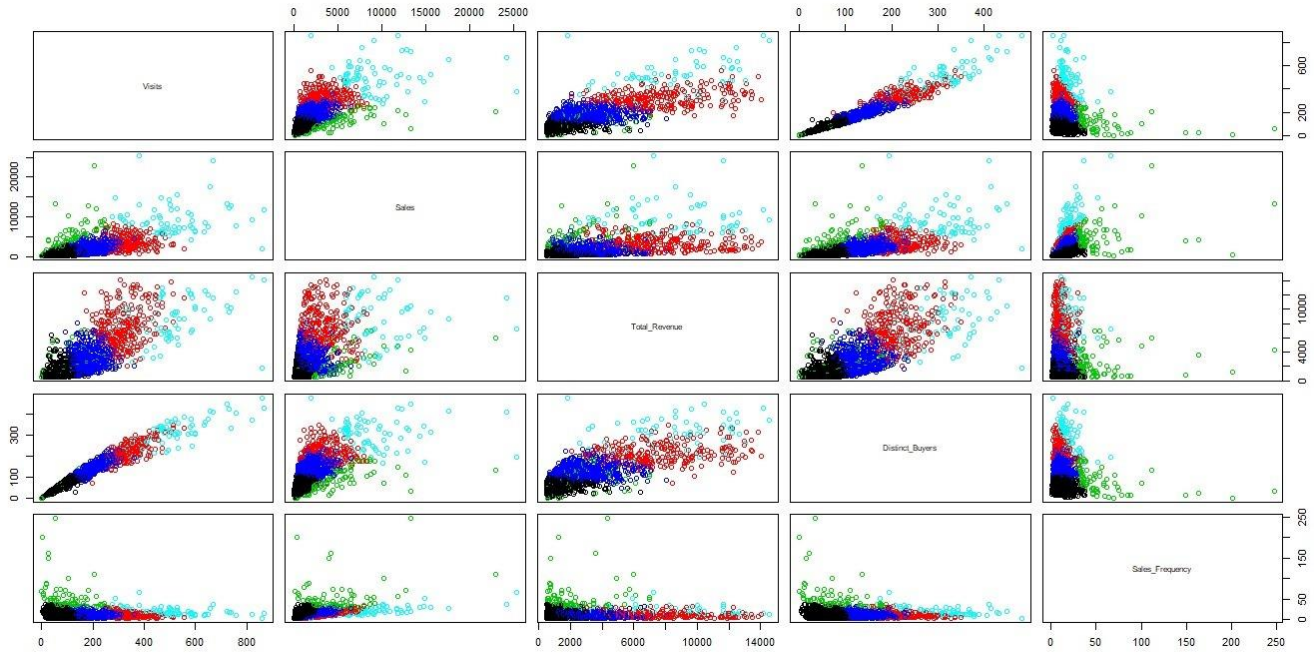


Fig: Formation of 5 clusters for Products dataset.

Cluster 1- **Black**

Cluster 2-Red

Cluster 3- Green

Cluster 4 - Blue

Cluster 5- Aqua blue

```
> Product.realCenters
      Visits     Sales Total_Revenue Distinct_Buyers Sales_Frequency
1  70.76653  702.3543      1185.956         56.06198         10.37930
2 319.69262 3370.4590      7616.523        213.77049         10.91374
3 110.75309 4902.7531      2296.949         84.40741         52.96077
4 177.16603 1838.3893      2930.803        129.83015         10.50054
5 510.88060 9336.7313      8439.771        310.37313         19.69625
```

Fig: Centroid for the clusters of the Product dataset

```
> pkm$centers
      Visits       Sales Total_Revenue Distinct_Buyers Sales_Frequency
1 -0.6626043 -0.49937963   -0.58637838      -0.6906222      -0.1698262
2  1.4198946  0.66006384    1.73570823       1.4364551      -0.1297546
3 -0.3280795  1.32593291   -0.18519762      -0.3083162       3.0228591
4  0.2275267 -0.00570772    0.04368805       0.3043184      -0.1607356
5  3.0193603  3.25274906    2.03298416       2.7393734       0.5287428
> pkm$totss
[1] 9415
> pkm$withinss
[1] 551.0444 581.9661 878.9654 499.6502 463.5396
> pkm$tot.withinss
[1] 2975.166
> pkm$betweenss
[1] 6439.834
> pkm$size
[1] 968 244  81 524  67
> pkm$iter
[1] 5
```

Fig: Summary of k means for Product dataset

7.2. Customer Clustering

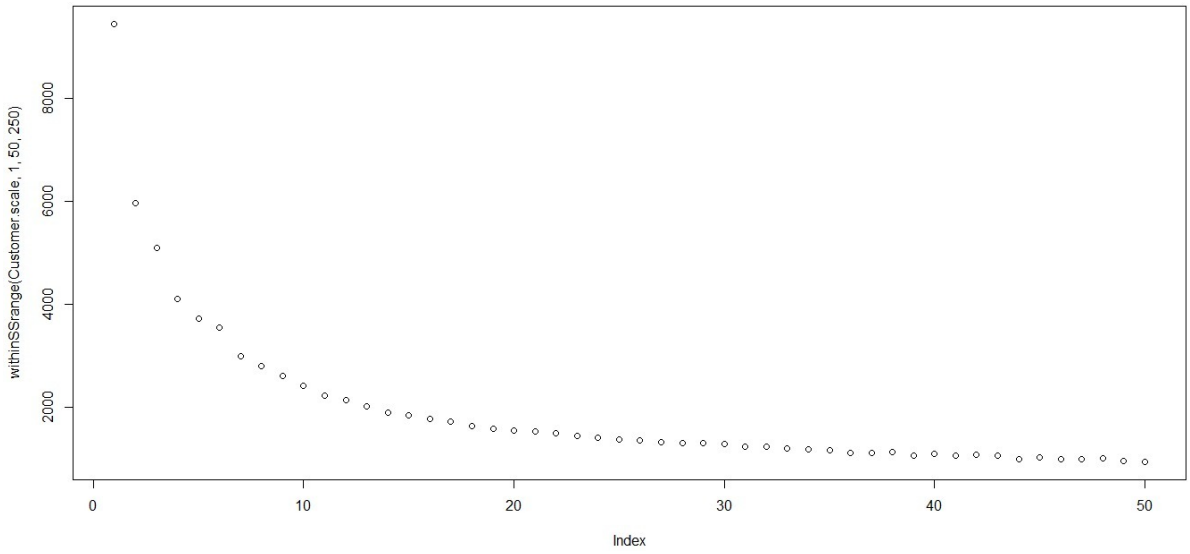
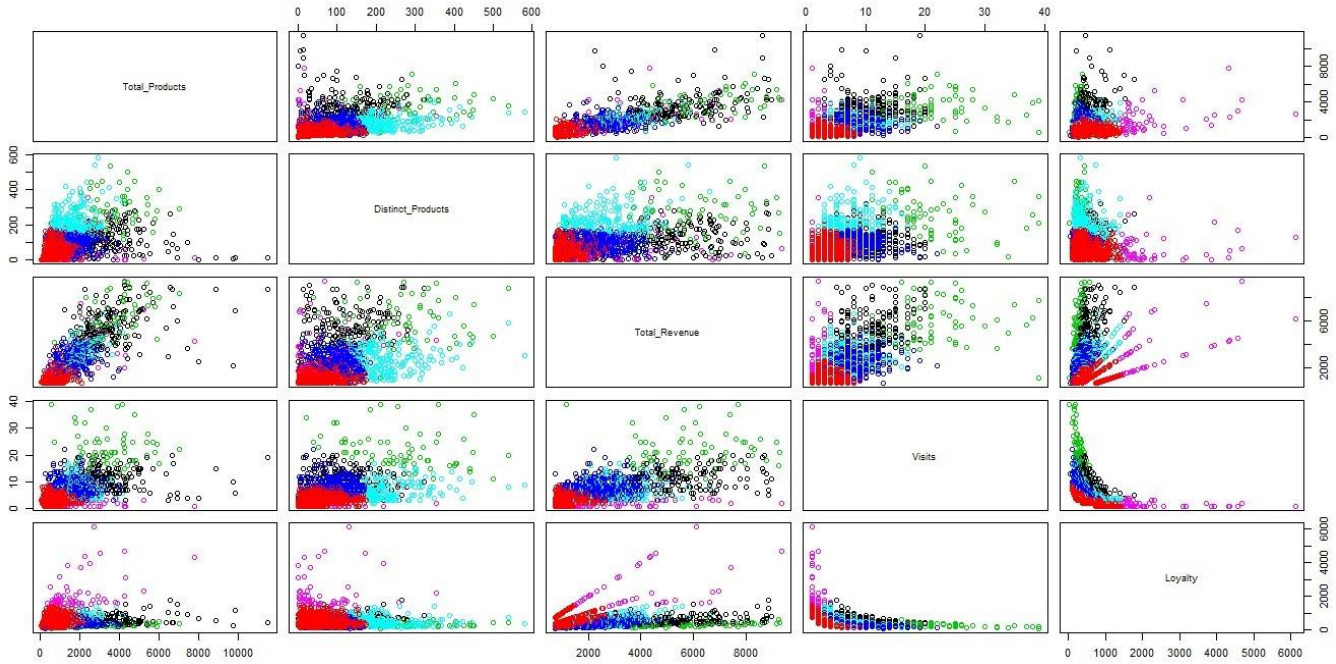


Fig: Elbow point for Customer dataset denotes that there should be 6 clusters



Cluster 1- **Black**

Cluster 2-Red

Cluster 3- Green

Cluster 4 - Blue

Cluster 5- Aqua blue

Cluster 5- Pink

Fig: Formation of 6 clusters for Customer dataset.

```
> Customer.realCenters
  Total_Products Distinct_Products Total_Revenue    Visits   Loyalty
1      3807.8014          124.95205      5667.322 11.335616  570.0173
2       717.6609           56.51629      1221.391  3.389002  440.6763
3      3497.8955          257.65672      5946.330 22.074627  290.7188
4      1497.2957           92.77826      2485.068  7.686957  360.0096
5      1739.1957          229.49457      2835.329  6.760870  505.7637
6      2055.9434           65.60377      3362.738  1.584906 2251.9985
```

Fig: Centroid for the clusters of the Customer dataset

```
> pkm$centers
  Total_Products Distinct_Products Total_Revenue     Visits     Loyalty
1     2.07058039        0.38969915     2.0651718  1.1154614  0.20686763
2    -0.56592528       -0.49519038    -0.6263161 -0.5415862 -0.09992677
3     1.80616894        2.10559978     2.2340786  3.3547863 -0.45562287
4     0.09925854       -0.02631512     0.1386917  0.3546339 -0.29126638
5     0.30564745        1.74145693     0.3507333  0.1615239  0.05445932
6     0.57589640       -0.37768725     0.6700173 -0.9177808  4.19649380
> pkm$totss
[1] 9455
> pkm$withinss
[1] 671.9272 760.9305 434.8105 544.5554 484.8409 502.1888
> pkm$tot.withinss
[1] 3399.253
> pkm$betweenss
[1] 6055.747
> pkm$size
[1] 146 982  67 460 184  53
> pkm$iter
[1] 3
```

Fig: Summary of k means for Product dataset

8. Cluster Profiling

According to the clusters formed above we deep dive into the details further provided by the datasets in forms of charts, clusters and their centroid. We describe and analyze both Product and Customer clusters separately.

8.1. Product Clusters

Rows contained by each cluster

Cluster	Description	Recommendation
One- **Better let go**	-low on sales, rarely bought, just stocked on -Not much contribution	Invest less and try to finish the remaining stock with low price.
Two- **Could Join the Hype**	-Bought Often, bought during various visits -contribute to revenue -need a little attention to be the bestseller	Promotions, discounts, devise marketing strategies, Pay more attention and stock often
Three- **Seasonal Hype**	-Used to be bought frequently	-Upgrade the seasonal products

	but lost touch -Seasonal products or once in a while statement -May be Outdated electronic devices	as well as the electronics if they are outdated -Keep track of trends and change of taste of your customers
Four- **Consistent but need attention**	- Sold very less - Have maintained consistency over time	-Frequently check upon i, maintain quality or it may fall in Cluster one -Offer complimentary products to increase interest of the customers
Five- **On the Hype**	-Among the best sellers -Few but contribute to increase the revenue -Attract new customers -Maybe gadgets or fashion products	-Monitor the sales constantly so as to maintain the sales balance. -Organize a sale for increasing number of customers -lower the price at times to attract more visitors.

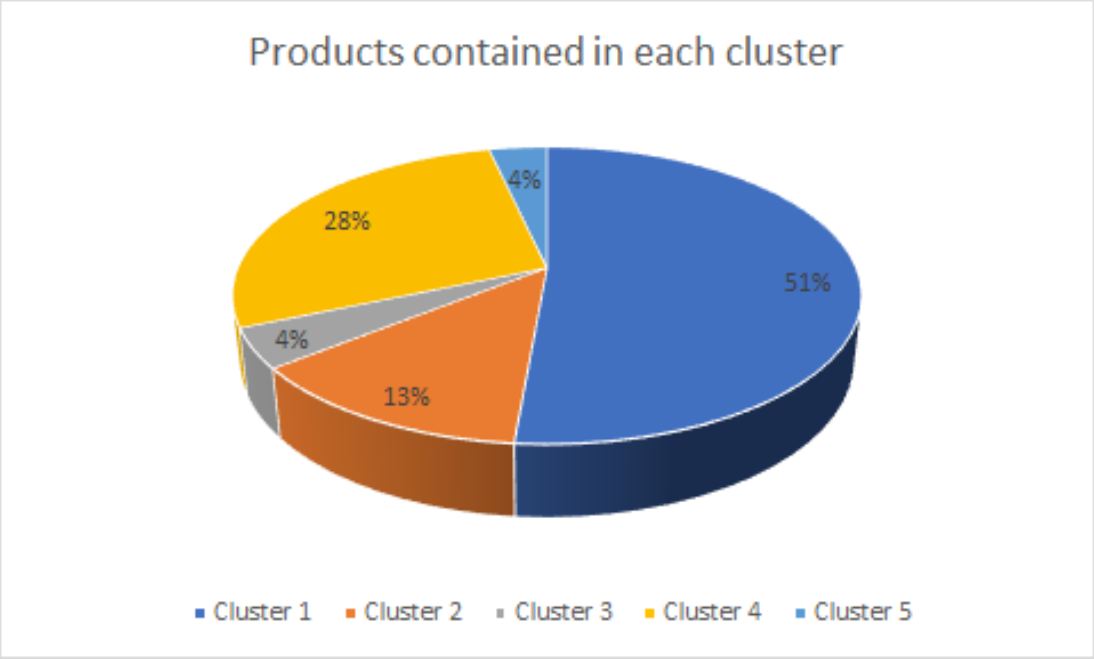


Fig: Products contained in each cluster

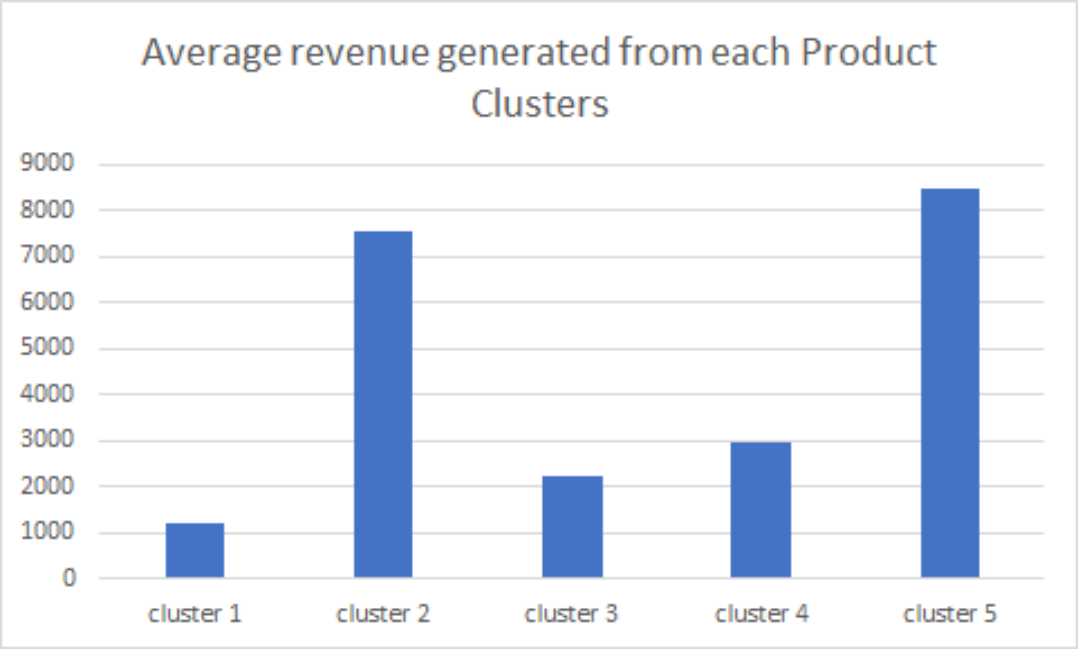


Fig: Average Revenue showing cluster 5 products generating highest revenue

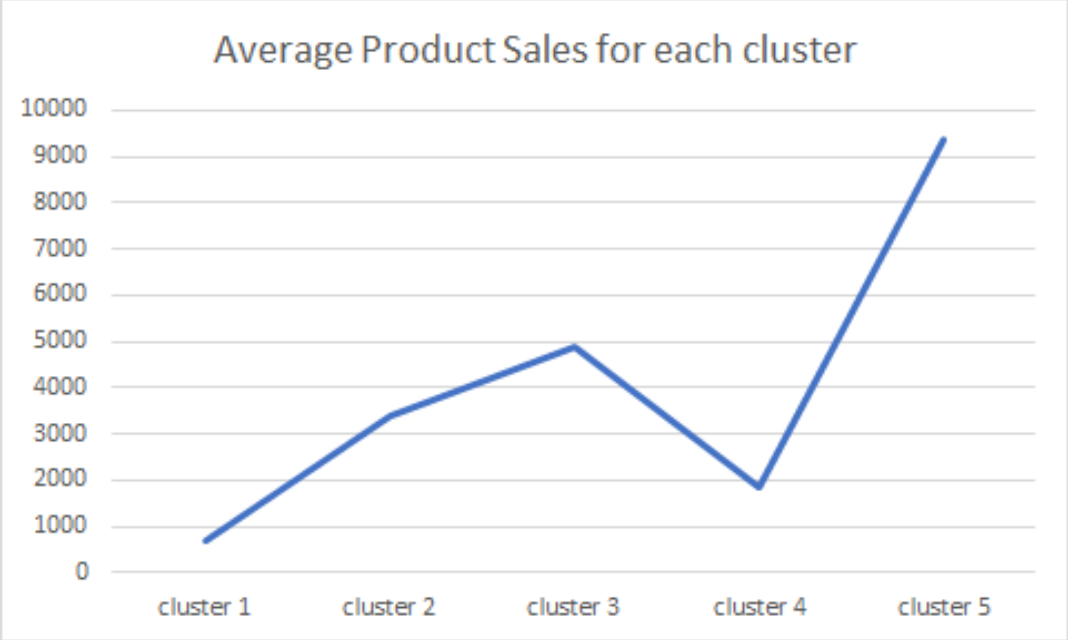


Fig: Average Product sale showing cluster 1 products having lowest sale

8.2. Customer Clusters

Cluster	Description	Recommendation
One- **Loyal Customer**	- Visit Often, Have contributed a lot of revenue - The ones who buy cheap groceries as well as expensive jewellery.	-Provide frequent assistance, greetings. -Give Privileges so that they maintain their loyalty -Festival discounts/casual discounts
Two- **Rare Customer**	-Visit rarely and buy rarely -not much difference to the revenue	- Provide more options for multiple products, provide products of various price range from cheap to expensive.
Three-**Best Customers**	-Visit a lot and buy a lot -Adding to the revenue -Buy expensive products	-Give Privileges so that they continue -always remember to keep track of and restock their favourite item. -Keep on changing the products according to trends
Four-**Consistent customers**	-Visit and buy sometime	Targeted promotions and sales and interesting deals that would make them spend
Five- **Good customers**	-Visit and buy every now and then -Don't buy as much expensive things but buy a whole lot of things , maybe groceries and smaller items	- Give more attention to their needs, giving complementary products on random visits.
Six- **Bargain Customers**	-They visit once in a while but rarely buy anything	-Provide refund policy and provide variety of price range.

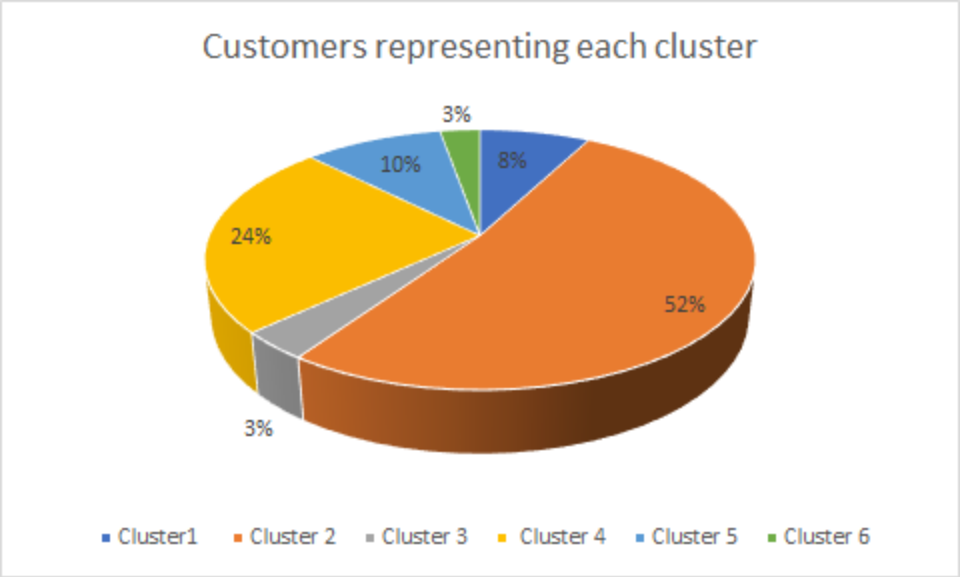


Fig: Customers representing each cluster

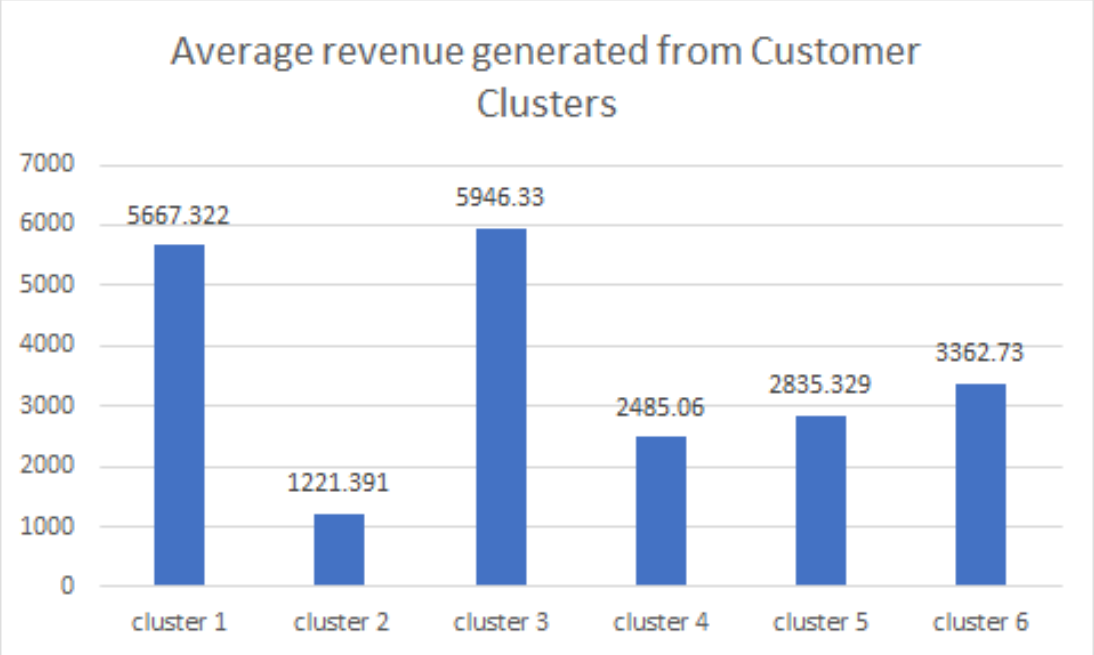


Fig: Average Revenue showing cluster 5 customers generating highest revenue

9. Conclusion & Next steps

Hereby, we divided the dataset into various clusters and analyzed for profiling both product and customer clusters. We addressed analytical questions like which product we can try to increase the price to add more revenue and for which product we need cut-off the price to attract more users to buy it. We not only get the data from this analysis but can also convert it to real business strategy.

Furthermore, we could have worked on the timings and recency for further investigation as we mentioned. Rather than unsupervised k-means which is only applicable in the datasets we can cluster we could have used decision trees which is more of a supervised learning. We can conclude from this analysis that the company is not doing that good as a lot of customers are in the margin of being a rare one or a standard one rather than a loyal customer. Similarly, a lot of products are lacking market and proper exposure and it also seems that the company is not following proper trends in selling various kinds of products which it should work upon to save its market value. Overall, the performance of the company is average and can perform better if recommended suggestions are taken over seriously. A lot of further analysis can be done on the basis of existing attributes to derive another conclusion.

10. References

[1] Optimove. (2019). *RFM Segmentation, Analysis & Model | Optimove*. [online] Available at: https://www.optimove.com/learning-center/rfm-segmentation [Accessed 24 May 2019].

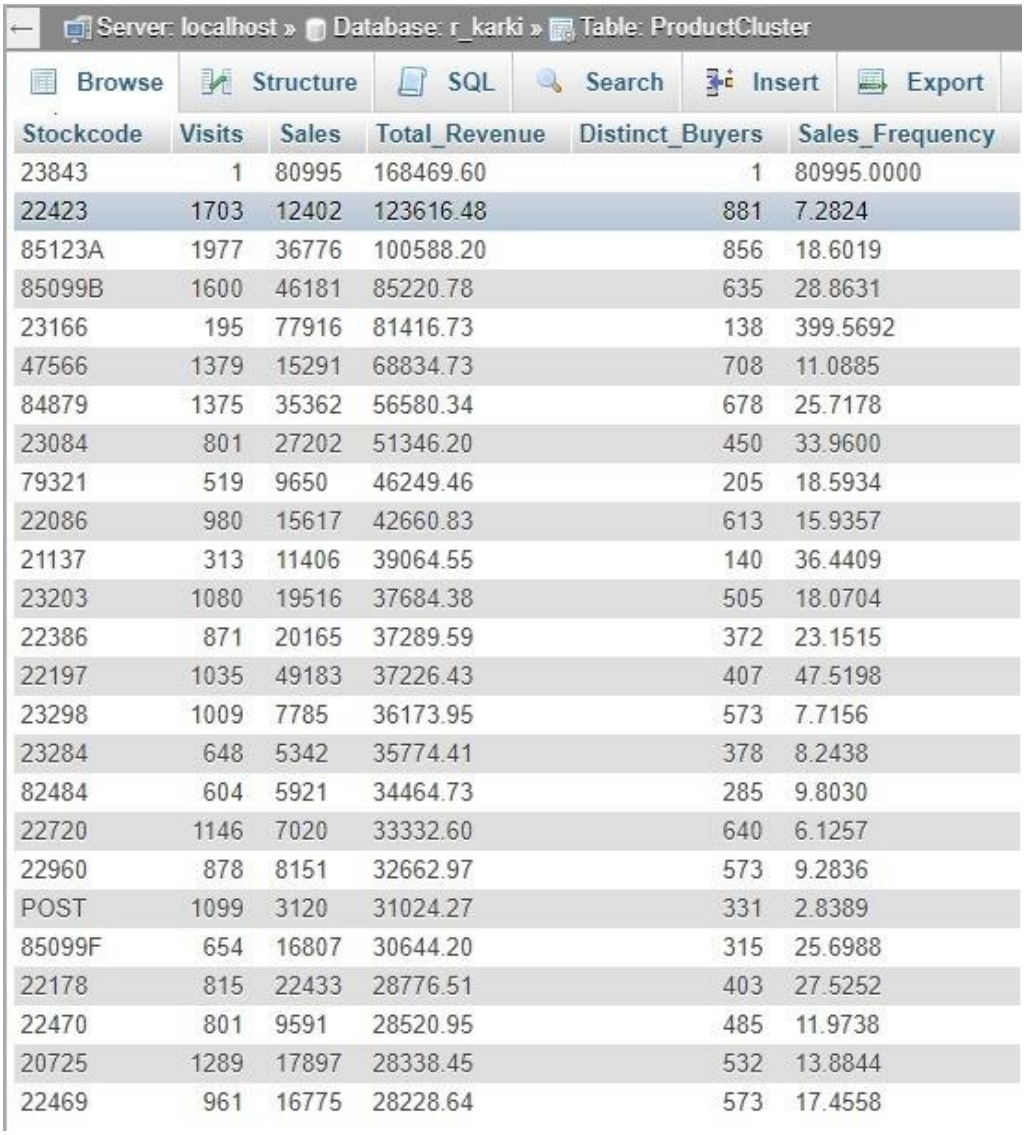
[2] Rpubs.com. (2019). *RPubs - Removing outliers - quick & dirty*. [online] Available at: http://rpubs.com/Mentors_Ubiqum/removing_outliers [Accessed 24 May 2019].

11. [Appendix A] SQL Query

11.1. SQL query for Product table

For Product Table

```
#SQL script for creating ProductCluster table
CREATE TABLE ProductCluster
AS
SELECT Stockcode,
COUNT(DISTINCT InvoiceNo) AS Visits,
SUM(Quantity) AS Sales,
SUM(Quantity*UnitPrice) AS Total_Revenue,
COUNT(DISTINCT CustomerID) AS Distinct_Buyers,
SUM(Quantity)/Count(DISTINCT InvoiceNo) AS Sales_Frequency
FROM dataset04.OnlineRetail
WHERE InvoiceNo!="0" AND CustomerID!="0" AND UnitPrice!=0
GROUP BY Stockcode
ORDER BY Total_Revenue DESC
LIMIT 0,2000;
```



Stockcode	Visits	Sales	Total_Revenue	Distinct_Buyers	Sales_Frequency
23843	1	80995	168469.60	1	80995.0000
22423	1703	12402	123616.48	881	7.2824
85123A	1977	36776	100588.20	856	18.6019
85099B	1600	46181	85220.78	635	28.8631
23166	195	77916	81416.73	138	399.5692
47566	1379	15291	68834.73	708	11.0885
84879	1375	35362	56580.34	678	25.7178
23084	801	27202	51346.20	450	33.9600
79321	519	9650	46249.46	205	18.5934
22086	980	15617	42660.83	613	15.9357
21137	313	11406	39064.55	140	36.4409
23203	1080	19516	37684.38	505	18.0704
22386	871	20165	37289.59	372	23.1515
22197	1035	49183	37226.43	407	47.5198
23298	1009	7785	36173.95	573	7.7156
23284	648	5342	35774.41	378	8.2438
82484	604	5921	34464.73	285	9.8030
22720	1146	7020	33332.60	640	6.1257
22960	878	8151	32662.97	573	9.2836
POST	1099	3120	31024.27	331	2.8389
85099F	654	16807	30644.20	315	25.6988
22178	815	22433	28776.51	403	27.5252
22470	801	9591	28520.95	485	11.9738
20725	1289	17897	28338.45	532	13.8844
22469	961	16775	28228.64	573	17.4558

Fig: Table CustomerCluster created after removing the negative and null values

11.2. SQL query for Product table

For Customers Table

```
#SQL script for creating CustomerCluster table
CREATE TABLE CustomerCluster AS
SELECT CustomerID,SUM(Quantity) AS Total_Products,
COUNT(DISTINCT StockCode) AS Distinct_Products,
SUM(Quantity*UnitPrice) AS Total_Revenue,
COUNT(DISTINCT InvoiceNo) AS Visits,
SUM(Quantity*UnitPrice)/COUNT(DISTINCT InvoiceNo) AS Loyalty
FROM dataset04.OnlineRetail
WHERE CustomerID!="0" AND InvoiceNo!="0" AND UnitPrice!=0
GROUP BY CustomerID
ORDER BY TOTAL_REVENUE DESC
LIMIT0, 2000;
```

Browse | Structure | SQL | Search | Insert | Export | Imp

CustomerID	Total_Products	Distinct_Products	Total_Revenue	Visits	Loyalty
14646	196915	700	277335.75	73	3799.119863
18102	64124	150	251614.30	60	4193.571667
17450	69993	124	194069.03	46	4218.891957
16446	80997	3	168472.50	2	84236.250000
14911	80265	1787	134452.38	201	668.917313
12415	77374	444	123594.12	21	5885.434286
14156	57885	714	113246.45	55	2059.026364
17511	64549	453	90939.50	31	2933.532258
12346	74215	1	77183.60	1	77183.600000
16029	40208	44	72892.08	63	1157.017143
16684	50255	119	66324.76	28	2368.741429
13694	63312	366	64945.06	50	1298.901200
15311	38194	567	60692.22	91	666.947473
17949	30546	29	57784.72	45	1284.104889
13089	31070	636	57754.42	97	595.406392
15769	29672	26	56252.72	26	2163.566154
15061	28920	70	53299.58	48	1110.407917
14298	58343	884	51320.42	44	1166.373182
14096	16352	1119	50366.93	17	2962.760588
14088	12665	379	48692.85	13	3745.603846
15749	18028	5	44534.30	3	14844.766667
12931	28004	28	42055.96	15	2803.730667
17841	23071	1323	39579.55	124	319.189919
13798	23948	113	37153.85	57	651.821930
16013	15536	30	36884.84	47	784.783830

Fig: Table CustomerCluster created after removing the negative and null values

12.[Appendix B] R scripts

12.1 Product Cluster R Script

```
#Load All Required Libraries
library(ggplot2)
library(GGally)
library(DMwR)

#Set the seed of R's random number generator
set.seed(5500)

#Read CSV file
Product <- read.csv("C:/Users/Tushar Mahat/Downloads/Rishi Downloads/Data
Mining/Assignment 1/ProductCluster.csv")

#View the loaded dataset
View (Product)

#Visualize data before outlier removal
ggpairs(Product[, which(names(Product) != "Stockcode")], upper = list(continuous =
ggally_points),lower = list(continuous = "points"), title = "Products before outlier removal")

boxplot(Product$Total_Revenue)

Q1 <- quantile(Product$Total_Revenue, probs = 0.25)
Q3 <- quantile(Product$Total_Revenue, probs = 0.75)
IQR <- Q3-Q1
minimum <- Q1-3*IQR
maximum <- Q3+3*IQR
Product.clean <- Product[-which(Product$Total_Revenue>maximum),]

ggpairs(Product.clean[, which(names(Product.clean) != "Stockcode")], upper =
list(continuous = ggally_points),lower = list(continuous = "points"), title = "Products after
removal of outliers")


Product.clean <- Product.clean[-which(Product.clean$Sales>40000),]
Product.clean <- Product.clean[-which(Product.clean$Sales_Frequency>500),]
```

```
boxplot(Product.clean$Total_Revenue)

#Visualize data after outlier removal
ggpairs(Product.clean[, which(names(Product.clean) != "Stockcode")], upper =
list(continuous = ggally_points),lower = list(continuous = "points"), title = "Products after
removal of outliers")

#Normalize data using scale and exclude CustomerID column. -1 will remove first column
that is CustomerID and keep all other.
Product.scale = scale(Product.clean[-1])

#View normalize data
View(Product.scale)

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
}

#Elbow plot to determine the optimal number of clusters between 1 and 50
plot(withinSSrange(Product.scale,1,50,150))

#K-means using k=6 for products based on results of  elbow plot
pkm = kmeans(Product.scale, 6, 150)

pkm$centers
pkm$totss
pkm$withinss
pkm$tot.withinss
pkm$betweenss
pkm$size
pkm$iter

#Denormalize data by reversing scale function
Product.realCenters = unscale(pkm$centers, Product.scale)
```

```
#Bind clusers to cleaned Data
ClusteredProduct = cbind(Product.clean, pkm$cluster)

View(ClusteredProduct)

#Visualize clustering results
plot(ClusteredProduct[,2:6], col=pkm$cluster)

#Export result to file
write.csv(ClusteredProduct,file="C:/Users/Tushar Mahat/Downloads/Rishi Downloads/Data
Mining/Assignment 1/ProductClusterResult.csv", col.names = FALSE)
```

12.2 Customer Cluster R Script

```
#Load All Required Libraries
library(ggplot2)
library(GGally)
library(DMwR)

#Set the seed of R's random number generator
set.seed(5500)

#Read CSV file
Customer <- read.csv("C:/Users/Tushar Mahat/Downloads/Rishi Downloads/Data
Mining/Assignment 1/CustomerCluster.csv")

#View the loaded dataset
View (Customer)

#Visualize data before outlier removal
ggpairs(Customer[, which(names(Customer) != "CustomerID")], upper = list(continuous =
ggally_points),lower = list(continuous = "points"), title = "Customers before outlier
removal")

boxplot(Customer$Total_Revenue)

Q1 <- quantile(Customer$Total_Revenue, probs = 0.25)
Q3 <- quantile(Customer$Total_Revenue, probs = 0.75)
IQR <- Q3-Q1
minimum <- Q1-3*IQR
maximum <- Q3+3*IQR
Customer.clean <- Customer[-which(Customer$Total_Revenue>maximum),]

Customer.clean <- Customer.clean[-which(Customer.clean$Visits>40),]
Customer.clean <- Customer.clean[-which(Customer.clean$Total_Products>15000),]

boxplot(Customer.clean$Total_Revenue)

#Visualize data after outlier removal
ggpairs(Customer.clean[, which(names(Customer.clean) != "CustomerID")], upper =
list(continuous = ggally_points),lower = list(continuous = "points"), title = "Customers after
removal of outliers")
```

```
#Normalize data using scale and exclude CustomerID column. -1 will remove first column
that is CustomerID and keep all other.
Customer.scale = scale(Customer.clean[-1])

#View normalize data
View(Customer.scale)

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
}

#Elbow plot to determine the optimal number of clusters between 1 and 50
plot(withinSSrange(Customer.scale,1,50,250))

#K-means using k=5 for customers based on results of  elbow plot
pkm = kmeans(Customer.scale, 5, 250)

pkm$centers
pkm$totss
pkm$withinss
pkm$tot.withinss
pkm$betweenss
pkm$size
pkm$iter

#Denormalize data by reversing scale function
Customer.realCenters = unscale(pkm$centers, Customer.scale)

#Bind clusers to cleaned Data
ClusteredCustomer = cbind(Customer.clean, pkm$cluster)

View(ClusteredCustomer)

#Visualize clustering results
```

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


#Export result to file
write.csv(ClusteredCustomer,file="C:/Users/Tushar Mahat/Downloads/Rishi
Downloads/Data Mining/Assignment 1/CustomerClusterResults.csv", col.names = FALSE)
```