CAPSTONE PROJECT

Customer Segmentation using K-Means Clustering

APRIL 15, 2022

P SANJAY NAIDU
Imarticus Learning

Table Of Contents

Summary of Dataset ->

The main aim of this project is to identify different segments of customers based on common characteristics or patterns. Segmentation of customers can have many factors, based on demographic, personal interest, behaviour or a combination of characteristics. By analysing customer purchase history, we can categorize them into similar groups and use those insights to make decisions for targeted marketing strategies.

For this particular project we are going to use K-mean Clustering which is an essential clustering algorithm. We are going to use this dataset and format the data in a way that the algorithm can process and implement the changes to determine the different customer segments.

This Project is a part of the First Capstone Project done under the PGPA course provided by Imarticus Learning. The dataset list purchases made by customers over a period of one year (from 2010/12/01 to 2011/12/09).

Transforming Data and Exploratory analysis was conducted on the data using R Studio, a language and a software environment for statistical computing.

- We see most of the variables are character datatype and we need to change them to appropriate datatype before working on them.
- The Quantity and Unit Price have a minimum negative value which needs to be checked.

```
> summary(cust_data)
  InvoiceNo          StockCode          Description
 Length:541909      Length:541909      Length:541909
 Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character


    Quantity         InvoiceDate          UnitPrice
 Min.   :-80995.00   Length:541909      Min.   :-11062.06
 1st Qu.:      1.00   Class :character   1st Qu.:      1.25
 Median :      3.00   Mode  :character   Median :      2.08
 Mean   :      9.55                      Mean   :      4.61
 3rd Qu.:     10.00                      3rd Qu.:      4.13
 Max.   :  80995.00                      Max.   :  38970.00

   CustomerID       Country
 Min.   :12346     Length:541909
 1st Qu.:13953     Class :character
 Median :15152     Mode  :character
 Mean   :15288
 3rd Qu.:16791
 Max.   :18287
 NA's   :135080
```

Required Libraries ->

These are the libraries which were used for the desired project to gain insights and aid us in better analysis for the customer segmentation.

```
#Loading Required Libraries
library(ggplot2)
library(dplyr)
library(tidyr)
library(DataExplorer)
library(lubridate)
library(heatmaply)
library(dlookr)
library(highcharter)
library(factoextra)
library(scales)
library(ClusterR)
library(cluster)
library(caret)
```

Loading Dataset ->

The Dataset consists of 541,909 transactions by 4,373 unique customers. Each customer is represented by a CustomerID and customers have multiple transactions. The dataset is loaded locally as a CSV file which was downloaded from Kaggle.

#E-COMMERCE DATASET:

#HTTPS://WWW.KAGGLE.COM/FABIENDANIEL/CUSTOMER-SEGMENTATION/DATA

Preview of the dataset is as below:

```
> head(cust_data)
  InvoiceNo StockCode                          Description Quantity    InvoiceDate UnitPrice CustomerID
1    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6 12/1/2010 8:26      2.55      17850
2    536365     71053                 WHITE METAL LANTERN        6 12/1/2010 8:26      3.39      17850
3    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8 12/1/2010 8:26      2.75      17850
4    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6 12/1/2010 8:26      3.39      17850
5    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6 12/1/2010 8:26      3.39      17850
6    536365     22752        SET 7 BABUSHKA NESTING BOXES        2 12/1/2010 8:26      7.65      17850
         Country
1 United Kingdom
2 United Kingdom
3 United Kingdom
4 United Kingdom
5 United Kingdom
6 United Kingdom
```

The following is a brief description of the data.

• **InvoiceNo:** A 6-digit number uniquely assigned to each transaction. The letter C indicates a cancellation.

• **StockCode:** A 5-digit number that is uniquely assigned to each product.

• **Description:** Product name.

• **Quantity:** The quantities of each product per transaction.

• **InvoiceDate:** Invoice date and time. Includes the day and time when a transaction was generated.

• **UnitPrice:** Product price per unit.

• **CustomerID:** A 5-digit number uniquely assigned to each customer.

• **Country:** The name of the customer's country.

We see three data types: character, integer, and number.

```
> str(cust_data)
'data.frame':   541909 obs. of  8 variables:
 $ InvoiceNo  : chr  "536365" "536365" "536365" "536365" ...
 $ StockCode  : chr  "85123A" "71053" "84406B" "84029G" ...
 $ Description: chr  "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUPID HEARTS COAT
R" "KNITTED UNION FLAG HOT WATER BOTTLE" ...
 $ Quantity   : int  6 6 8 6 6 2 6 6 6 32 ...
 $ InvoiceDate: chr  "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26" ...
 $ UnitPrice  : num  2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...
 $ CustomerID : int  17850 17850 17850 17850 17850 17850 17850 17850 17850 13047 ...
 $ Country    : chr  "United Kingdom" "United Kingdom" "United Kingdom" "United Kingdom" ...
```

Transforming Data ->

➢ *Treating Missing Values:*

We find that we have a large chunk of missing of missing values in CustomerID column. We see that these entries are not assigned to any particular customer.

```
> colSums(is.na(cust_data))
  InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice  CustomerID     Country
          0           0           0           0           0           0      135080           0
```

Now we are going to drop these rows having NA values as these do not bode well in building a successful model. We are going to do this by using the complete.cases function.

```
> dim(cust_data)
[1] 406829      8
```

With complete.cases function we are not dropping the entries but selecting only those entries which are complete i.e. not have any missing values.

We see that after using the following treatment we are still left with 406829 cases which is more than sufficient to further our assessment.

➢ *Treating Outliers:*

We check for the presence of Outliers and we notice that maximum products which are sold are low priced. Some of the Outliers present can be cancelled or wrong orders which were assigned negative value.

We are using plot_outliers function in order to plot and view them. We do this for both Quantity and Unit Price.

```
plot_outlier(cust_data, Quantity, col = "steelblue")
plot_outlier(cust_data, UnitPrice, col = "steelblue")
```

We are treating the outliers with help of a loop and assigning them to NA.

```
#Treating Ouliers
for (x in c('Quantity','UnitPrice'))
{
  value = cust_data[,x][cust_data[,x] %in% boxplot.stats(cust_data[,x])$out]
  cust_data[,x][cust_data[,x] %in% value] = NA
}
```

We see we have NA values generated again as we treating outliers now.

```
> colSums(is.na(cust_data))
  InvoiceNo   StockCode Description    Quantity InvoiceDate   UnitPrice  CustomerID     Country
          0           0           0       26682           0       36051           0           0
```

Now we are going to change the negative values as well to NA and finally drop them.

➢ *Negative Values:*

We see that Quantity and Unit Price are the most important in the dataset. We also see that both the variables have few negative values.

```
> summary(cust_data$Quantity)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
  -13.00     2.00     4.00     6.82    12.00    27.00    26682
> summary(cust_data$UnitPrice)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
    0.00     0.85     1.65     2.14     2.95     7.50    36051
```

These negative values in the dataset are because of the cancelled orders which are indicated with the letter C before the InvoiceNo. We also see NA's because of presence of Outliers.

```
> head(neg_quantity, 5)
  InvoiceNo StockCode                         Description Quantity     InvoiceDate UnitPrice CustomerID
1   C581484     23843         PAPER CRAFT , LITTLE BIRDIE   -80995  12/9/2011 9:27      2.08      16446
2   C541433     23166      MEDIUM CERAMIC TOP STORAGE JAR   -74215 1/18/2011 10:17      1.04      12346
3   C536757     84347 ROTATING SILVER ANGELS T-LIGHT HLDR    -9360 12/2/2010 14:23      0.03      15838
4   C550456     21108  FAIRY CAKE FLANNEL ASSORTED COLOUR    -3114 4/18/2011 13:08      2.10      15749
5   C550456     21175         GIN + TONIC DIET METAL SIGN    -2000 4/18/2011 13:08      1.85      15749
         Country
1 United Kingdom
2 United Kingdom
3 United Kingdom
4 United Kingdom
5 United Kingdom
```

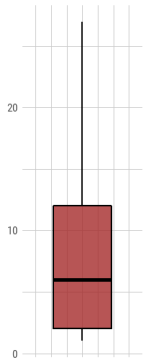
```
> cust_data <- cust_data %>%
+   mutate(Quantity = replace(Quantity, Quantity<=0, NA),
+          UnitPrice = replace(UnitPrice, UnitPrice<=0, NA))
> cust_data <- cust_data %>%
+   drop_na()
> dim(cust_data)
[1] 338151      8
```

Now we remove these negative values in Quantity and Unit Price by replacing them with NA. After doing so we are going to drop all the NA values from our dataset. We now remain with 338,151 observations which is a significant amount to analyse and build our model.

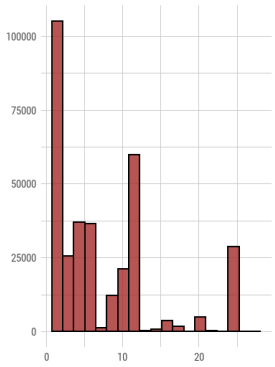
Here is the plot of the Outliers for Quantity and Unit Price.

Outlier Diagnosis Plot (Quantity)

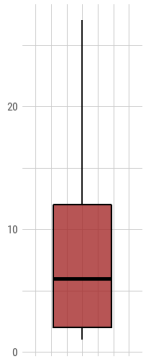
With outliers



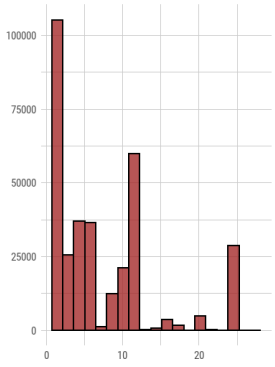
With outliers



Without outliers

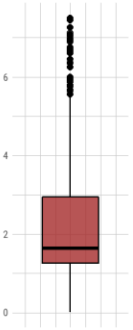


Without outliers

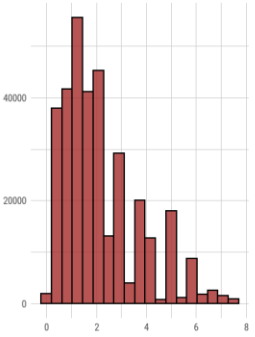


Outlier Diagnosis Plot (UnitPrice)

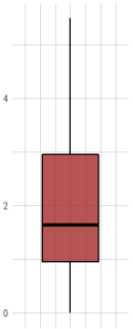
With outliers



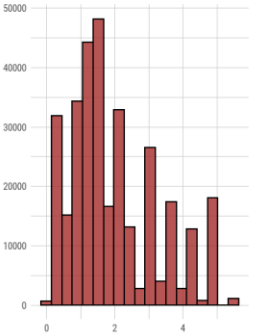
With outliers



Without outliers



Without outliers



➢ *Separating Date and Time component:*

We are going to separate Date and Time components from the InvoiceDate for further analysis. For this we use function(x) and strsplit() functions.

First, we separate Date and Time from InvoiceDate ->

```
cust_data$Date <- sapply(cust_data$InvoiceDate, FUN = function(x) {strsplit(x, split = '[ ]')[[1]][1]})
```

```
cust_data$Time <- sapply(cust_data$InvoiceDate, FUN = function(x) {strsplit(x, split = '[ ]')[[1]][2]})
```

Now we separate Month, Year from Date ->

```
cust_data$Month <- sapply(cust_data$Date, FUN = function(x) {strsplit(x, split = '[/]')[[1]][1]})
```

```
cust_data$Year <- sapply(cust_data$Date, FUN = function(x) {strsplit(x, split = '[/]')[[1]][3]})
```

Now we separate Hour from Time ->

```
cust_data$Hour <- sapply(cust_data$Time, FUN = function(x) {strsplit(x, split = '[:]')[[1]][1]})
```

We convert Date column to the Date format ->

```
cust_data$Date <- as.Date(cust_data$Date, "%m/%d/%Y")
```

After this we create a new column for the day of the week using wday function ->

```
cust_data$Week <- wday(cust_data$Date, label=TRUE)
```

Here is the structure of the dataset now:

```
> str(cust_data)
'data.frame':   397884 obs. of  15 variables:
 $ InvoiceNo  : chr  "536365" "536365" "536365" "536365" ...
 $ StockCode  : chr  "85123A" "71053" "84406B" "84029G" ...
 $ Description: chr  "WHITE HANGING HEART T-LIGHT HOLDER" "WHITE METAL LANTERN" "CREAM CUPID HEARTS COAT HANG
ER" "KNITTED UNION FLAG HOT WATER BOTTLE" ...
 $ Quantity   : int  6 6 8 6 6 2 6 6 6 32 ...
 $ InvoiceDate: chr  "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26" "12/1/2010 8:26" ...
 $ UnitPrice  : num  2.55 3.39 2.75 3.39 3.39 7.65 4.25 1.85 1.85 1.69 ...
 $ CustomerID : int  17850 17850 17850 17850 17850 17850 17850 17850 13047 ...
 $ Country    : Factor w/ 37 levels "Australia","Austria",..: 35 35 35 35 35 35 35 35 35 35 ...
 $ Date       : Date, format: "2010-12-01" "2010-12-01" "2010-12-01" ...
 $ Time       : chr  "8:26" "8:26" "8:26" "8:26" ...
 $ Month      : Factor w/ 12 levels "1","10","11",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ Year       : Factor w/ 2 levels "2010","2011": 1 1 1 1 1 1 1 1 1 1 ...
 $ Hour       : Factor w/ 15 levels "10","11","12",..: 14 14 14 14 14 14 14 14 14 14 ...
 $ Week       : Ord.factor w/ 7 levels "Sun"<"Mon"<"Tue"<..: 4 4 4 4 4 4 4 4 4 4 ...
 $ Cost       : num  15.3 20.3 22 20.3 20.3 ...
```

➢ *Adding a new column:*

We add a new column “Cost” which is a product of Quantity and Unit Price.

```
#Adding a new column for Cost
cust_data <- cust_data %>% mutate(Cost = Quantity * UnitPrice)
```

➢ *Transforming the type of Variables:*

We are changing the variables into factors for further analysing of the dataset.

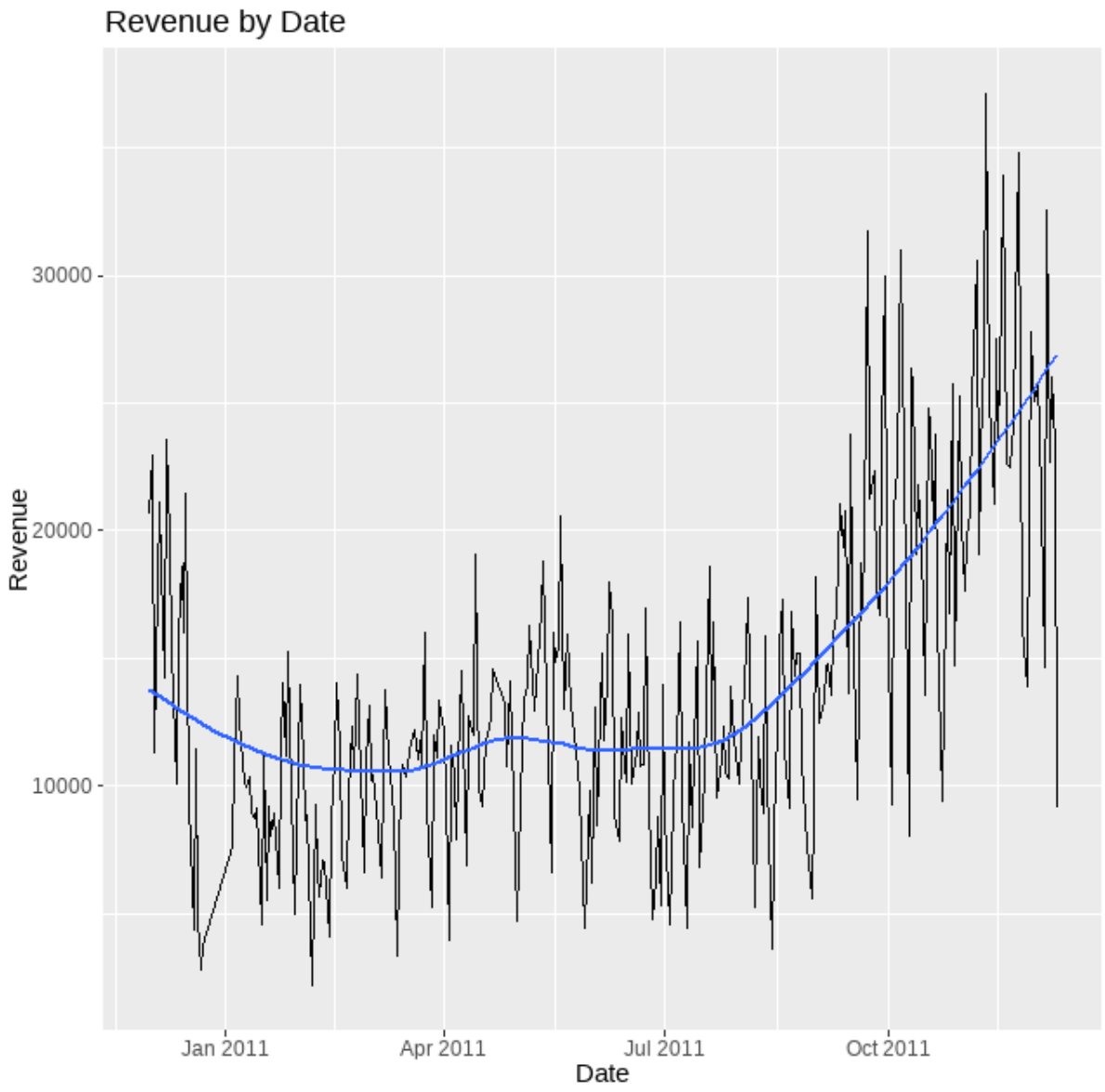
```
#We are changing the type of important variables into Factors
cust_data$Month <- as.factor(cust_data$Month)
cust_data$Year <- as.factor(cust_data$Year)
levels(cust_data$Year) <- c(2010,2011)
cust_data$Hour <- as.factor(cust_data$Hour)
cust_data$Week <- as.factor(cust_data$Week)
cust_data$Country <- as.factor(cust_data$Country)
```

Data Visualization ->

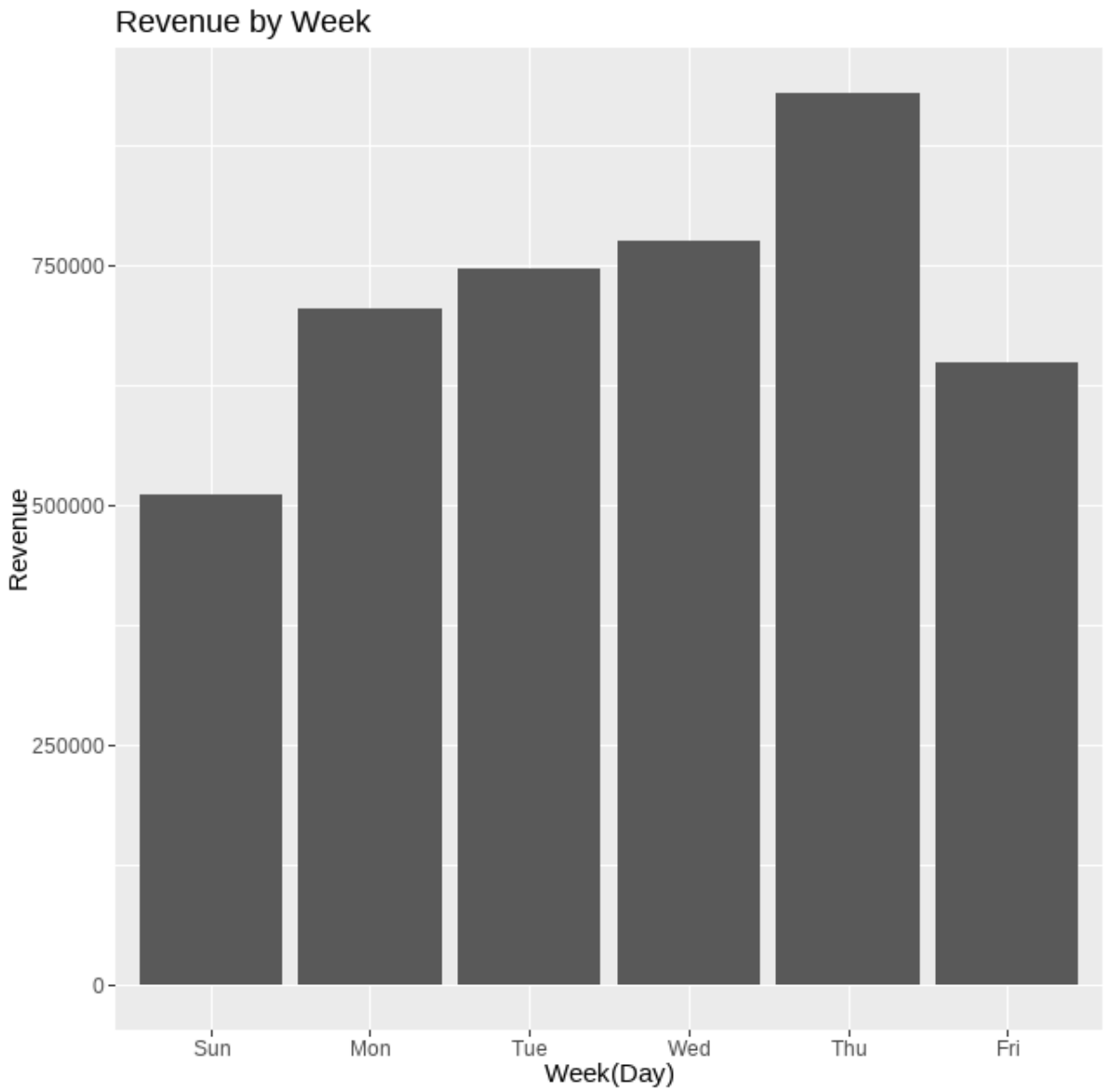
We now have better dataset to perform analysis. We are using EDA(Exploratory Data Analysis) to observe common patterns and spot anomalies. This is important because we need to understand our dataset before we start building our model.

➢ *Revenue Summary:*

We plot the Revenue to Date and see that there is a steady increase over time from September2011.



Second, we plot Revenue by Week for micro analysis.

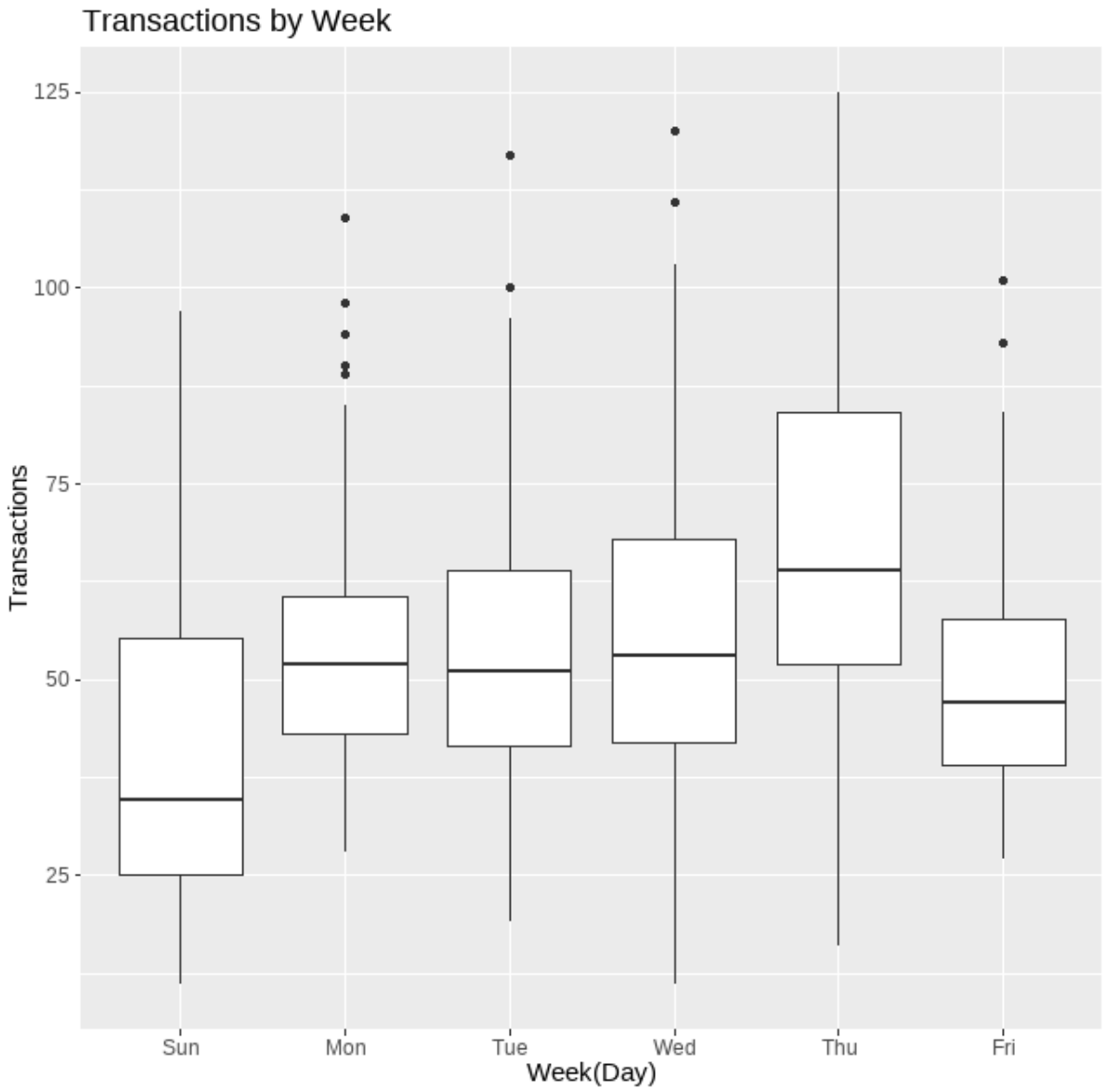


We see that most Revenues are generated during mid week with significant amount generated on Thursday and Sundays are generally low.

➢ *Transaction Summary:*

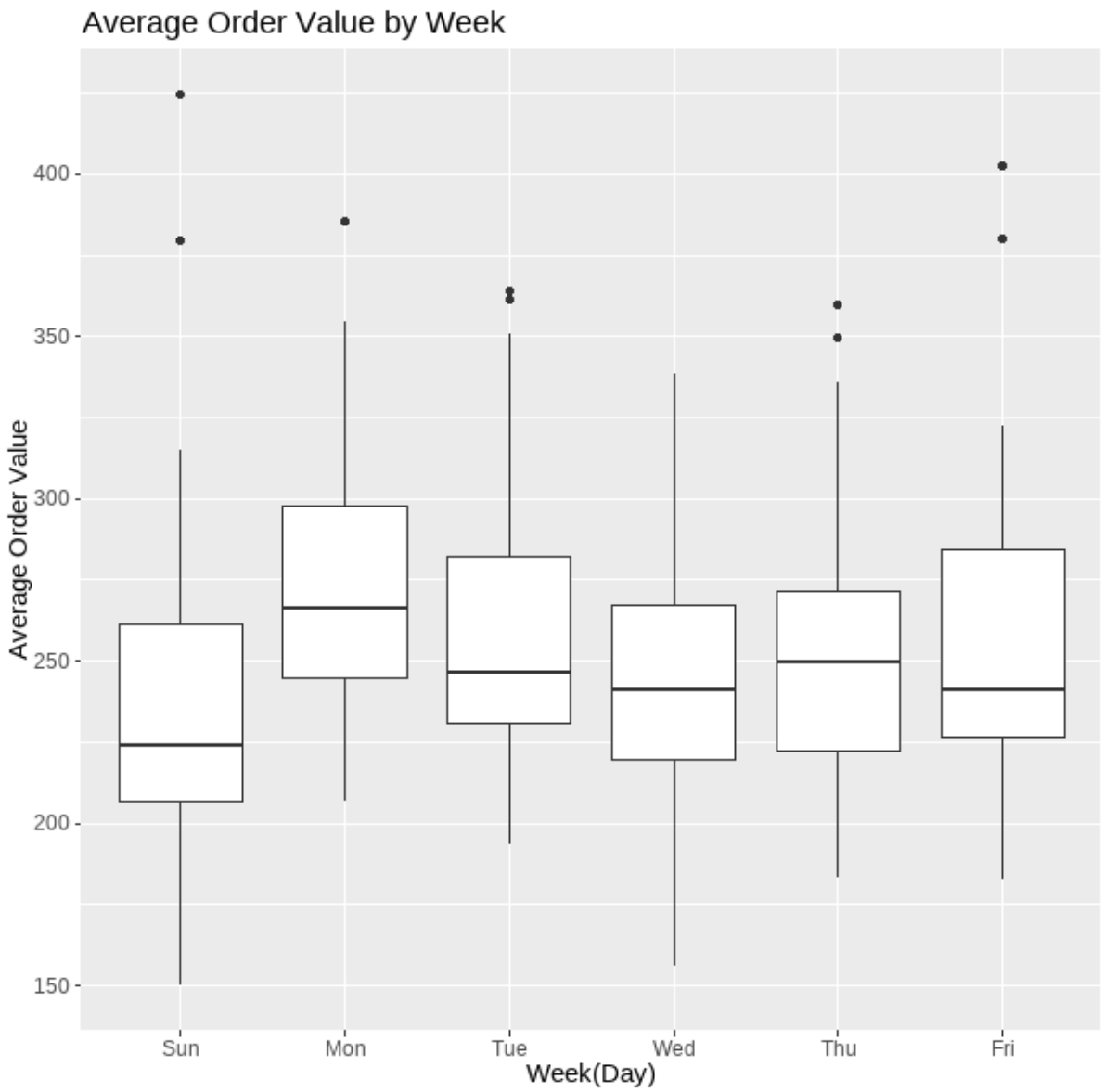
We are going to plot the Transactions vs Week plot. Before that we generate the revenues earned on weekly basis.

```
weekday_sum <- cust_data %>%
  group_by(Date, Week) %>%
  summarise(Revenue = sum(Cost), Transaction = n_distinct(InvoiceNo)) %>%
  mutate(Value = (round((Revenue / Transaction),2))) %>%
  ungroup()
```

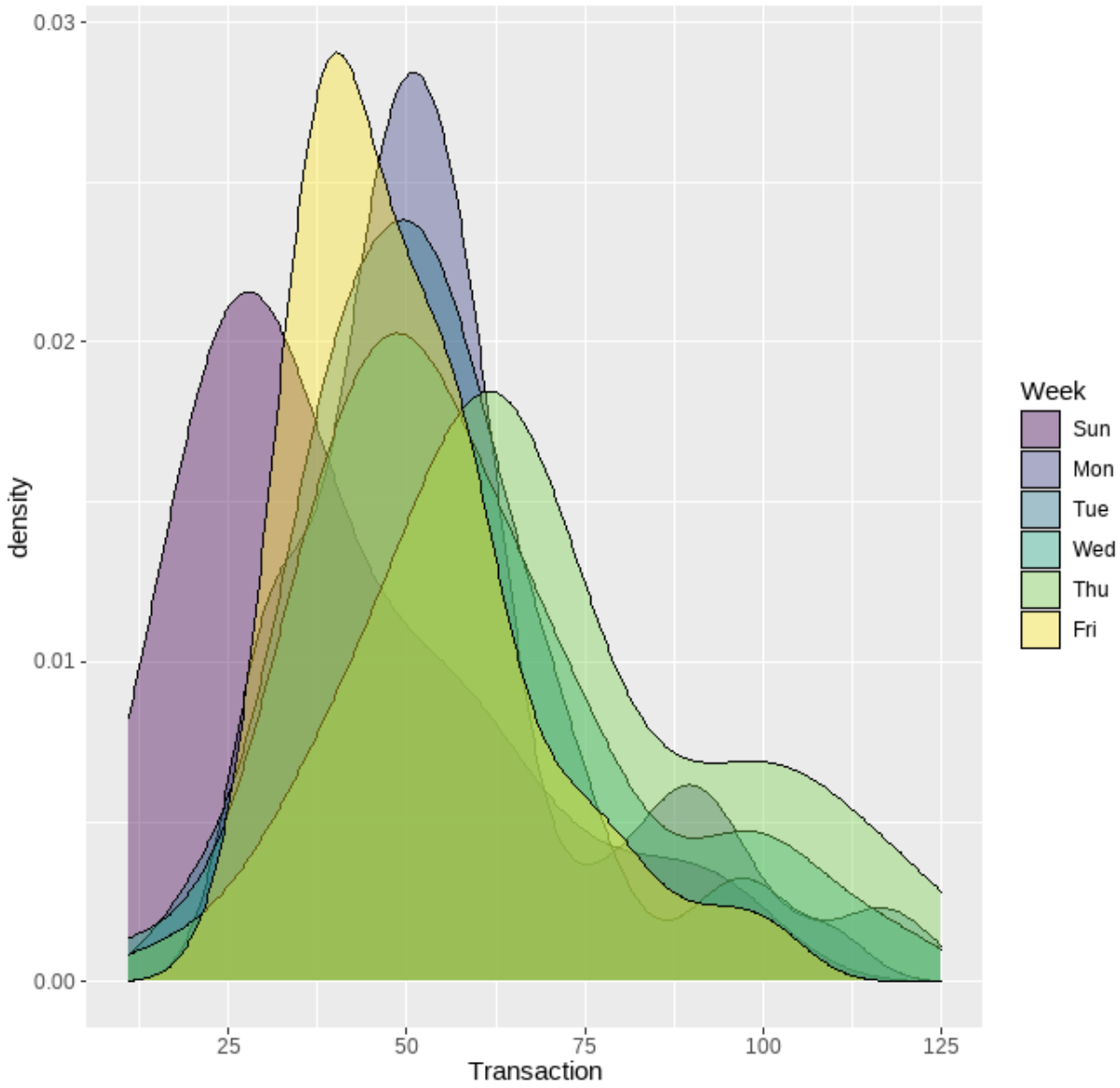


We can see that most transactions are made on Thursday and least transactions were on Sunday which is similar to the Revenue generated.

The difference in the amount of revenue generated on each day of the week is driven by a difference in the number of transactions, rather than the average order value as is evident in the chart below.



The chart below shows that there is a bit of skewness in our distribution with the least number of transactions leaning towards Sunday and Friday.

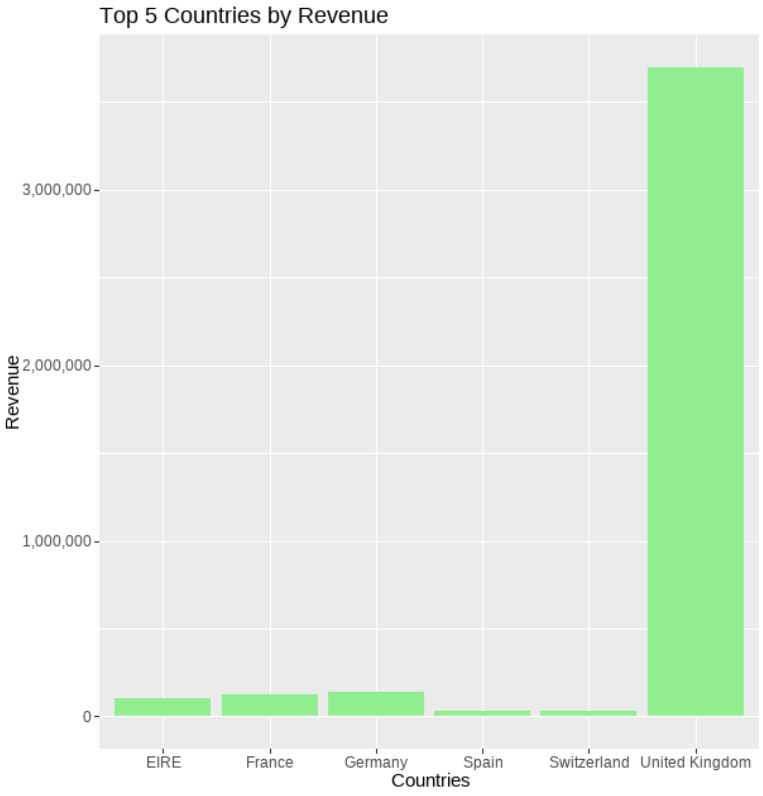

0.03
0.02
0.01
0.00
density
25
50
75
100
125
Transaction
Week
Sun
Mon
Tue
Wed
Thu
Fri

➢ *Country Summary:*

As per the below table we can see that the United Kingdom has the largest Revenue and Transactions whereas Netherlands has the least Revenue and the least Transaction belongs to Norway.

```
> head(SummaryofCountry, 10)
# A tibble: 10 x 4
   Country        Revenue Transaction Value
   <fct>            <dbl>       <int> <dbl>
 1 United Kingdom 3699806.       15144  244.
 2 Germany         138135.         425  325.
 3 France          122037.         362  337.
 4 EIRE            105962.         238  445.
 5 Spain            29846.          82  364.
 6 Switzerland      29790.          43  693.
 7 Belgium          27839.          93  299.
 8 Portugal         20463.          46  445.
 9 Norway           18123.          34  533.
10 Netherlands      15579.          65  240.
```

We will now plot and see the Revenues in Horizontal Bar Charts for Top 6 countries which is below:



We will remove United Kingdom and plot the chart again in order to remove bias.

Below is the table excluding United Kingdom denoting the same:

```
> top5
# A tibble: 5 x 5
  Country     Revenue Transaction Customers Value
  <fct>         <dbl>       <int>     <int> <dbl>
1 Germany     138135.         425        92  325.
2 France      122037.         362        87  337.
3 EIRE        105962.         238         3  445.
4 Spain        29846.          82        29  364.
5 Switzerland  29790.          43        21  693.
```

We see that Germany and France have the maximum Revenue followed closely by EIRE nations and the same norms apply for Transactions summary as well.

Here we end with our Visualization and now we create customer segments.

We are using CustomerID to look for differences between customers and summarize the results by revenue:

```
> head(cust_segmentation)
# A tibble: 6 x 4
  CustomerID Revenue Transaction Value
       <int>   <dbl>       <int> <dbl>
1      14911  85019.         193  441.
2      13089  36412.          80  455.
3      14096  36020.          17 2119.
4      17841  32645.         123  265.
5      14298  24388.          40  610.
6      12748  21047.         196  107.
```

We find that there are quite a lot of high-quantity sales and refunds as well and also, we see that many of the transactions are refunded as well.

We sum the revenue in order to work with some reasonable numbers.

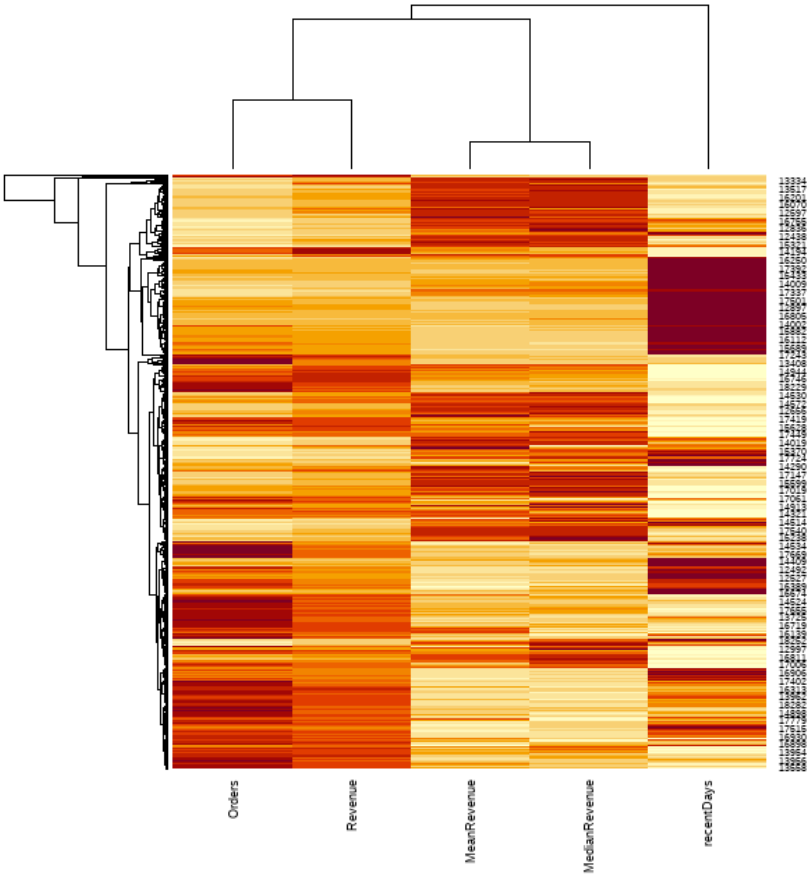
```
> head(cust_segmentation_2)
# A tibble: 6 x 6
  CustomerID InvoiceNo Revenue Transaction Value CummulativeSum
       <int> <chr>       <dbl>       <int> <dbl>          <dbl>
1      13405 568375      0.001           1  0             0.001
2      14800 570554      0.38            1  0.38          0.381
3      16669 567869      0.4             1  0.4           0.781
4      14744 542736      0.55            1  0.55          1.33
5      12748 548657      0.85            1  0.85          2.18
6      16554 540945      0.85            1  0.85          3.03
```

The below data frame provides us with the order value and date & time information for each transaction, that can be grouped by CustomerID. Along with this we also filter orders greater than 1 and Revenue greater than 50 pounds.

```
> head(cust_segmentation_summary, 5)
# A tibble: 5 x 9
  CustomerID Country  Orders Revenue MeanRevenue MedianRevenue MostDays MostHours recentDays
       <int> <fct>     <int>   <dbl>       <dbl>         <dbl> <chr>    <chr>          <int>
1      12347 Iceland       7  3315.        474.          352.  Tue      14              3786
2      12348 Finland       3    90.2        30.1          20.4 Tue      10              4032
3      12352 Norway        7  1131.        162.          178.  Tue      14              3820
4      12356 Portugal      2  1087.        543.          543.  Tue      12              4029
5      12358 Austria       2   878.        439.          439.  Tue      10              3785
```

We are now remaining with a small subset of 2693 observations putting us in a better position to answer questions about our customers which can be used to target specific marketing strategies.

By analysing the customers cluster, we discover groups of customers that behave in similar ways. This level of customer segmentation is useful in marketing to these groups of customers appropriately. A marketing campaign that works for a group of customers that place low value orders frequently may not be appropriate for customers who place sporadic, high value orders. We use a heat map to visualize the customers recent days, order and revenue scores. Higher scores are indicated by the darker areas in the heatmap.



Here the Recent Days represents number of days before the reference date when a customer made the last purchase. The lesser the value, higher is the likelihood of customer visiting the store.

Our algorithm aims to keep the distance between data points in a cluster as little as possible relative to the distance between two clusters. Members of separate groups are very distinct whereas individuals of one group are quite similar.

From the heatmap above, it is evident that the total revenue clusters with the number of orders as we would expect. The mean and median order values cluster together, again this is expected, and lastly the order recency sits in its own group. The significant point here is how the customers rows cluster. We are able to uncover groups of customers that behave in similar ways. We have an idea about the clusters, and we now proceed to employ K-means algorithm.

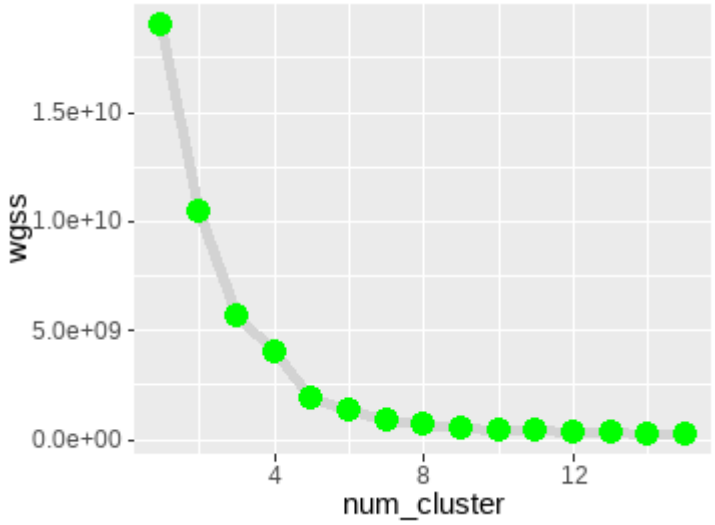
Steps Performed ->

- First, we loaded the data from the csv file.
- We then checked the structure and summary of the dataset.
- We found a chunk of missing values in the CustomerID section and treated it.
- We also found outliers which was treated by assigning the values to NA and later by dropping them.
- We also treated the negative values in Quantity and Unit Price by assigning them to NA and dropping them.
- We created a Cost column to ease our analysis.
- We then plotted the Revenue and Transaction by Week and further understood the dataset.
- We then analysed the top 5 Countries and excluded United Kingdom to remove the bias.
- We then created Customer Segments and filtered our data as per our convenience.
- We then applied the K-Means Clustering algorithm with suitable value of k.

Results ->

This section represents Segmentation approach (Using Elbow Method) that was employed and the results were obtained.

The main goal behind cluster partitioning methods like k-means is to define the clusters that maintain the intra-cluster variation at a minimum. The plot below denotes the appropriate number of clusters required in our model. In the plot, the location of a bend or a knee is the indication of the optimum number of clusters.



From the above graph we see the value of k as 4. We then attached these results to our Data Frame to identify each customer's cluster.

```
> head(dd)
      recentDays Revenue MeanRevenue MedianRevenue Orders cluster
12347       3786 3314.73      473.53        351.82      7       1
12348       4032   90.20       30.07         20.40      3       2
12352       3820 1130.94      161.56        177.83      7       4
12356       4029 1086.56      543.28        543.28      2       2
12358       3785  878.22      439.11        439.11      2       1
12359       3841 3224.13      806.03        721.54      4       3
```

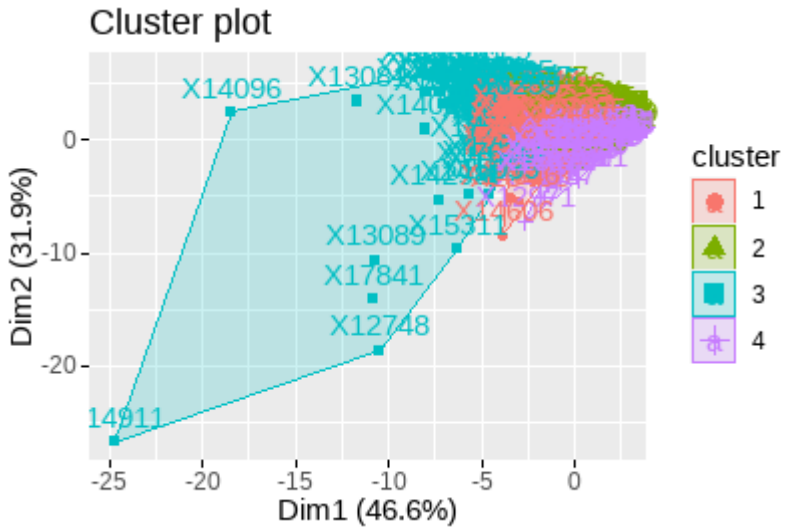
The algorithm divided our cluster sizes into 4 parts(k value) 785, 400, 88, 1420 respectively.

```
> km$size
[1]  785  400   88 1420
```

Below are the cluster centres:

```
> km$centers
  recentDays    Revenue MeanRevenue MedianRevenue      Orders
1 -0.2996484  0.3211648   0.7404585     0.7232604  0.10594671
2  2.0082633 -0.3717486  -0.4456964    -0.4096172 -0.34322436
3 -0.2974764  2.3847218   3.3060323     3.2519090  1.19470898
4 -0.3816221 -0.2206130  -0.4886705    -0.4859722 -0.03592452
```

Below is the Graphical representations of the 4 clusters:



It is evident from the results and the charts above that:

- Cluster 2 has mean Recent Days with 3981 but all the clusters are more or less near to each other.
- Cluster 3 has mean number of Orders with 15 customers.
- Mean Revenue is Highest for Cluster 3 with 853.1

Conclusion ->

We managed to identify two main segments of customers according to their revenue patterns, number of orders and Recent Days which will helps target the customers based on their habits. The other 2 cluster where more or less similar to each other in terms of Mean Revenues and Orders.

Through K-Means clustering were able to segment customers to get a better understanding of them which in turn could be used to increase a company's revenue. Other clustering algorithms such as Silhouette and Gap statistic methods could be used to identify the optimal number of clusters. For this project we restricted ourselves to the Elbow method only.

Further analysis using other segmentation approaches such as RFM Analysis, Principal Component Analysis (PCA) etc. can be conducted on the clusters to identify more narrowed characteristics of the customers, understand relationship between cluster and types of products purchased or predicting each cluster and customers lifetime value.