Customer Segmentation Project in R

ERICK@Guru

2024

Customer Segmentation Project in R

Customer Segmentation is one the most important applications of **unsupervisedMachine Learning**. Using **clustering** techniques, companies can identify the several segments of customers allowing them to target the potential user base. In this machine learning project, we will make use of ***K-means clustering.*** which is the essential algorithm for clustering **unlabeled dataset**. Before ahead in this project, learn what actually customer segmentation is??

What is Customer Segmentation?

Customer Segmentation *is the process of division of customer base into several groups of individuals that share a similarity in different ways that are relevant to marketing such as gender, age, interests, and miscellaneous spending habits.*

Companies that deploy customer segmentation are under the notion that every customer has different requirements and require a specific marketing effort to address them appropriately. Companies aim to gain a deeper approach of the customer they are targeting. Therefore, their aim has to be specific and should be tailored to address the requirements of each and every individual customer. Furthermore, through the data collected, companies can gain a deeper understanding of customer preferences as well as the requirements for discovering valuable segments that would reap them maximum profit. This way, they can strategize their marketing techniques more efficiently and minimize the possibility of risk to their investment.

The technique of customer segmentation is dependent on several key differentiators that divide customers into groups to be targeted. Data related to demographics, geography, economic status as well as behavioral patterns play a crucial role in determining the company direction towards addressing the various segments.

How to Implement Customer Segmentation in R?

In the first step of this data science project, we will perform data exploration. We will import the essential packages required for this role and then read our data. Finally, we will go through the input data to gain necessary insights about it.

```
#import libraries
library(tidyverse)
library(janitor)
library(ggplot2)
# library()
```

```
#import dataset
customer_data <-read_csv("Mall_Customers.csv") %>% clean_names()
names(customer_data)
```

```
## [1] "customer_id"          "gender"               "age"
## [4] "annual_income_k"      "spending_score_1_100"
```

```
#check duplicates rows
sum(duplicated(customer_data$customer_id))
```

```
## [1] 0
```

```
#data structure
glimpse(customer_data)
```

```
## Rows: 200
## Columns: 5
## $ customer_id          <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15~
## $ gender               <chr> "Male", "Male", "Female", "Female", "Female", "Fe~
## $ age                  <dbl> 19, 21, 20, 23, 31, 22, 35, 23, 64, 30, 67, 35, 5~
## $ annual_income_k      <dbl> 15, 15, 16, 16, 17, 17, 18, 18, 19, 19, 19, 19, 2~
## $ spending_score_1_100 <dbl> 39, 81, 6, 77, 40, 76, 6, 94, 3, 72, 14, 99, 15, ~
```

```
#convert gender to a factor variable
customer_data$gender <- factor(customer_data$gender)
levels(customer_data$gender)
```

```
## [1] "Female" "Male"
```

We will now display the first six rows of our dataset using the head() function and use the summary() function to output summary of it.

```
head(customer_data)
```

```
## # A tibble: 6 x 5
##   customer_id gender   age annual_income_k spending_score_1_100
##         <dbl> <fct>  <dbl>           <dbl>                <dbl>
## 1           1 Male      19              15                   39
## 2           2 Male      21              15                   81
## 3           3 Female    20              16                    6
## 4           4 Female    23              16                   77
## 5           5 Female    31              17                   40
## 6           6 Female    22              17                   76
```

```
#check summary
summary(customer_data[,-1]) #%>% DT::datatable()
```

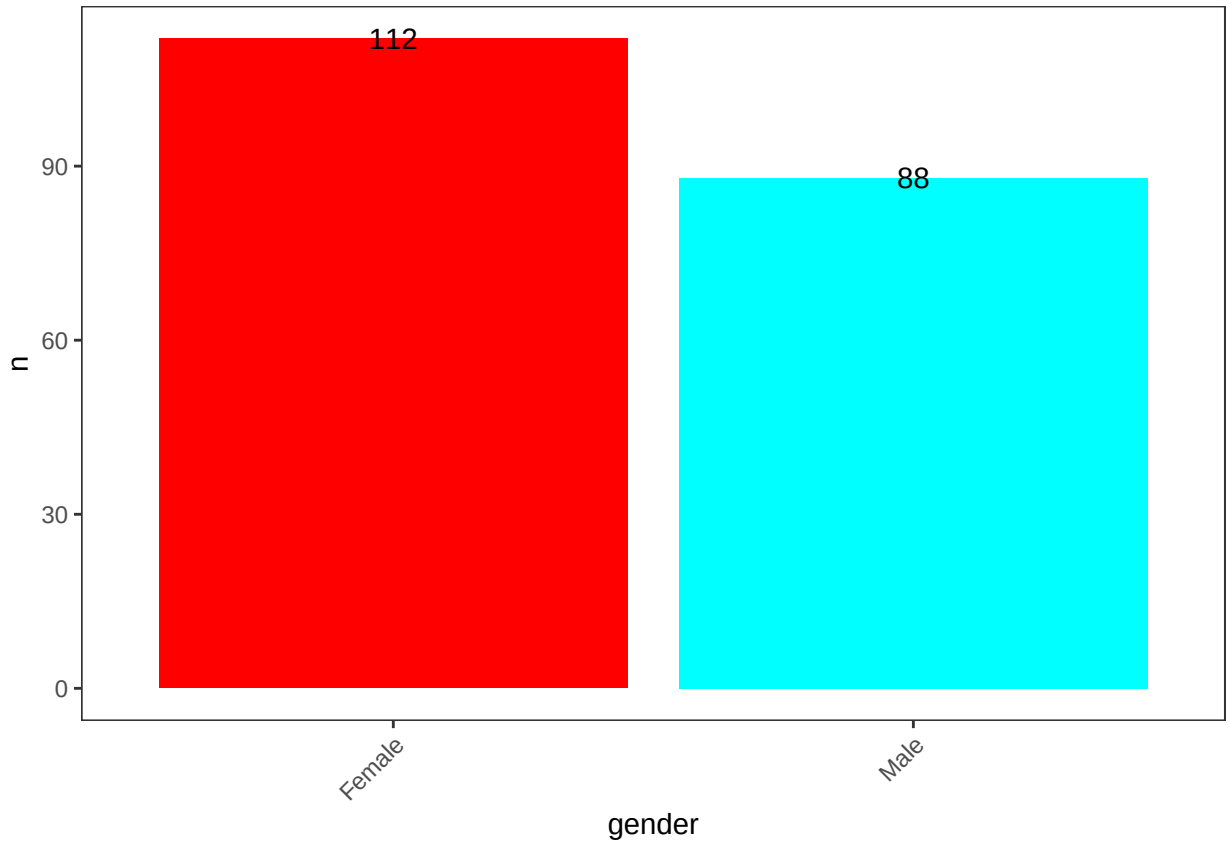
```
##     gender         age        annual_income_k  spending_score_1_100
##  Female:112   Min.   :18.00   Min.   : 15.00   Min.   : 1.00
##  Male  : 88   1st Qu.:28.75   1st Qu.: 41.50   1st Qu.:34.75
##               Median :36.00   Median : 61.50   Median :50.00
##               Mean   :38.85   Mean   : 60.56   Mean   :50.20
##               3rd Qu.:49.00   3rd Qu.: 78.00   3rd Qu.:73.00
##               Max.   :70.00   Max.   :137.00   Max.   :99.00
```

Customer Gender Visualization

In this, we will create a barplot and a piechart to show the gender distribution across our customer_data dataset.

```
# barplot(
#   table(customer_data$gender),
#   col = rainbow(2),
#  )
theme_set(theme_test())
plotdata <- customer_data %>% count(gender)

plotdata %>%
  ggplot(aes(x=gender, y=n)) +
  geom_bar(stat = 'identity', fill=rainbow(2))+
  geom_text(aes(label=n))+
  theme(axis.text.x = element_text(
    angle = 45,
    hjust = 1
  ))
```



From the above bar-plot, we observe that the **number of females** is **higher** than the **males**.

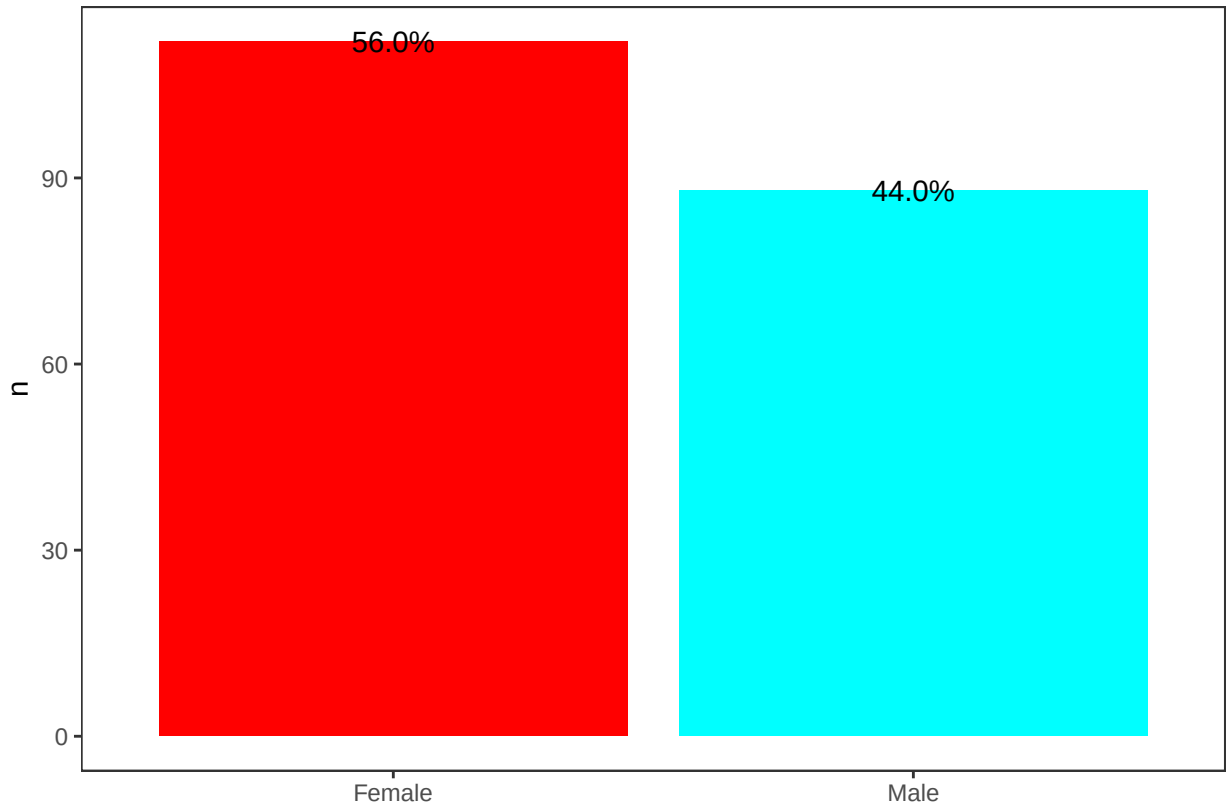
```
d <- customer_data %>% tabyl(gender) %>%
              adorn_pct_formatting() %>%
```

```
              as_tibble()
d
```

```
## # A tibble: 2 x 3
##   gender     n percent
##   <fct>  <dbl> <chr>
## 1 Female   112 56.0%
## 2 Male      88 44.0%
```

Now, let us visualize a chart to observe the ratio of male and female distribution.

```
d %>%
  ggplot(aes(x=gender, y=n))+
  geom_bar(stat = 'identity', fill=rainbow(2)) +
  geom_text(aes(label=percent))+
  #geom_text(aes(label=n))+
  xlab(" " )
```

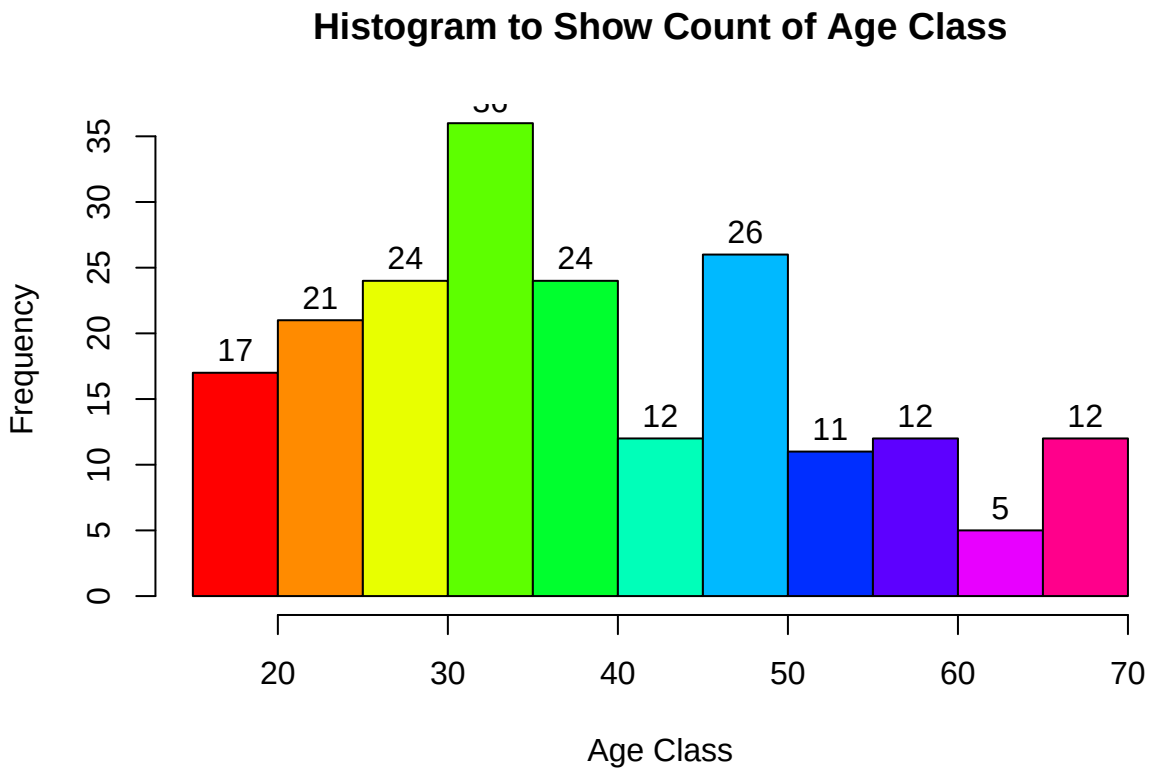


From the above graph, we conclude that the percentage of females is **56%**, whereas the percentage of male in the customer dataset is **44%**.

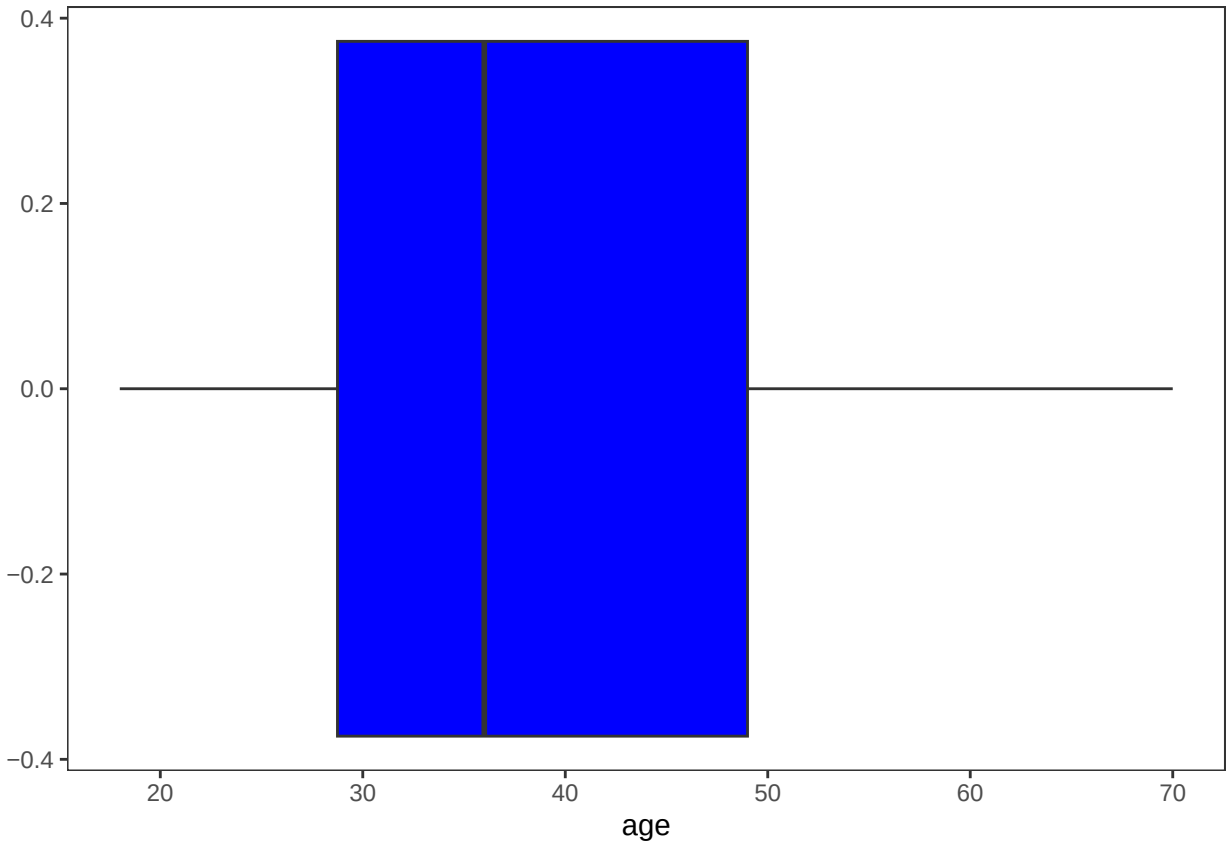
Visualization of Age Distribution

Let us plot a histogram to view the distribution to plot the frequency of customer ages. We will first proceed by taking summary of the Age variable.

```r
#Histogram of Ages
hist(customer_data$age,
    col=rainbow(11),
    main="Histogram to Show Count of Age Class",
    xlab="Age Class",
    ylab="Frequency",
    labels=TRUE)
```




```r
#barplot
# boxplot(customer_data$age,
#         col="deepskyblue",
#         main="Boxplot for Descriptive Analysis of Age")
# IQR(customer_data$age)
# quantile(customer_data$age, 1/4)
# quantile(customer_data$age, 2/4)
# quantile(customer_data$age, 3/4)
customer_data %>%
  ggplot(aes(age))+
  geom_boxplot(fill="blue")
```



From the above two visualizations, we conclude that the maximum customer ages are between 30 and 35. The minimum age of customers is 18, whereas, the maximum age is 70.

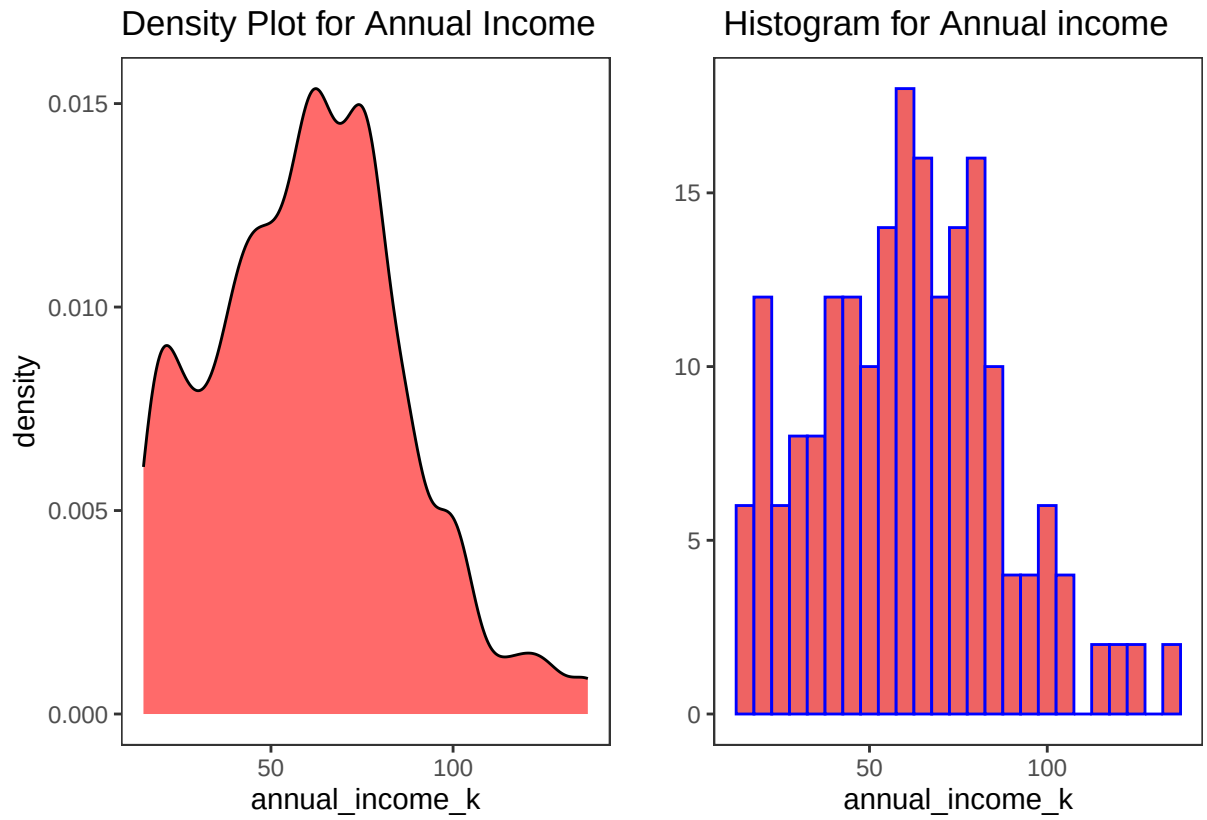
Analysis of the Annual Income of the Customers

In this section of the R project, we will create visualizations to analyze the annual income of the customers. We will plot a histogram and then we will proceed to examine this data using a density plot.

```
#Histogram
library(patchwork)
p1 <- customer_data %>%
  ggplot(aes(annual_income_k)) +
  geom_density(fill='indianred1',
               bw=5)+
  ggtitle('Density Plot for Annual Income')

p2 <- customer_data %>%
  ggplot(aes(annual_income_k))+
  geom_histogram(binwidth = 5,
                 fill='indianred2',
                 col='blue',
                 )+
  ylab('')+
  ggtitle(' Histogram for Annual income')

p1+p2
```



From the above descriptive analysis, we conclude that the minimum annual income of the customers is 15 and the maximum income is 137. People earning an average income of 70 have the highest frequency count in our histogram distribution. The average salary of all the customers is 60.56. In the Kernel Density Plot that we displayed above, we observe that the annual income has a normal distribution.

Analyzing Spending Score of the Customers

```
summary(customer_data$spending_score_1_100)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   34.75   50.00   50.20   73.00   99.00
```

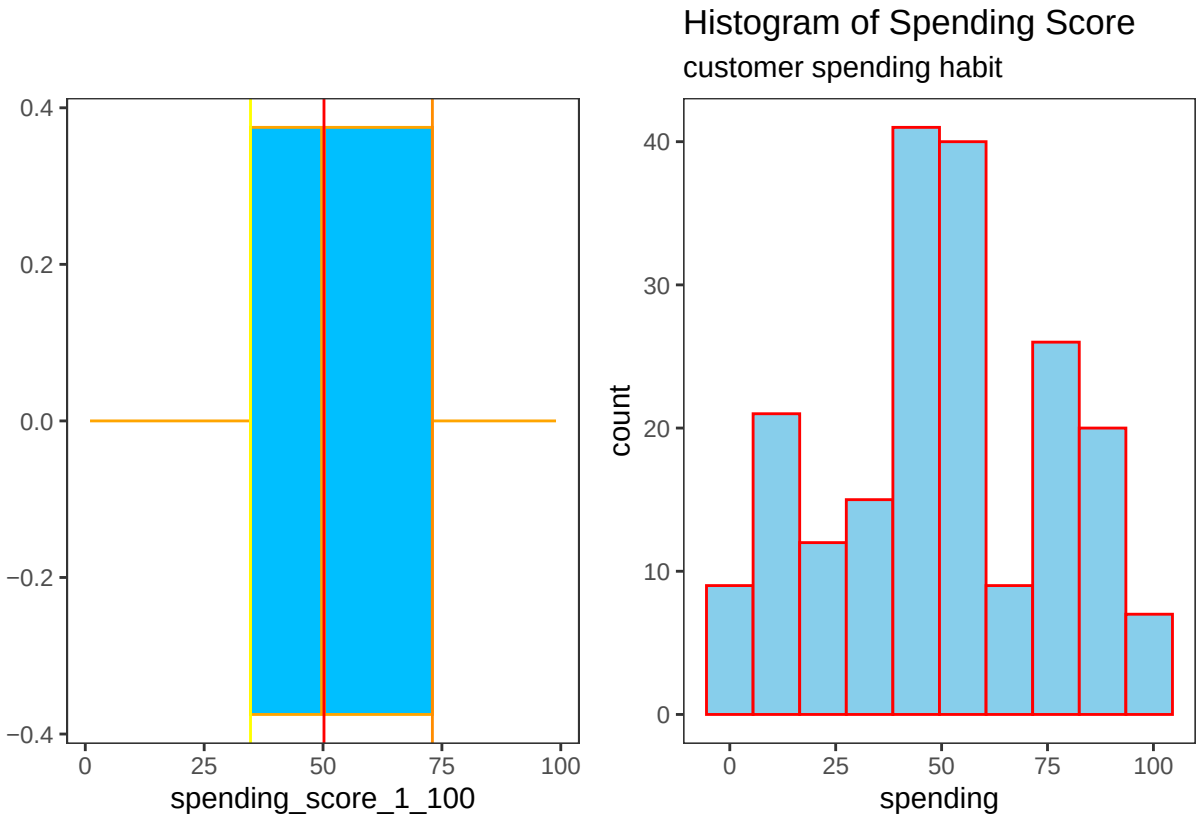
```
#boxplot

d1 <- customer_data %>%
  ggplot(aes(spending_score_1_100))+
  geom_boxplot(fill='deepskyblue',
               col='orange')+
  geom_vline(xintercept = 50.2,#mean.
             col='red')+
  geom_vline(xintercept =34.75,#1st Qu.
             col='yellow')+
  geom_vline(xintercept =73,#3rd Qu.
             col='darkorange'
```

```
                )
#Histogram

d2 <- customer_data %>%
  ggplot(aes(spending_score_1_100))+
  geom_histogram(binwidth=11,
                 fill='skyblue',
                 col='red')+
   labs(
     x='spending',
     y='count',
     title = 'Histogram of Spending Score',
     subtitle = 'customer spending habit'
   )

d1+d2
```



The minimum spending score is 1, maximum is 99 and the average is 50.20. We can see Descriptive Analysis of Spending Score is that Min is 1, Max is 99 and avg. is 50.20. From the histogram, we conclude that customers between class 40 and 50 have the highest spending score among all the classes.

K-means Algorithm

While using the k-means clustering algorithm, the first step is to indicate the number of clusters (k) that we wish to produce in the final output. The algorithm starts by selecting k objects from dataset randomly

that will serve as the initial centers for our clusters. These selected objects are the cluster means, also known as centroids. Then, the remaining objects have an assignment of the closest centroid. This centroid is defined by the Euclidean Distance present between the object and the cluster mean. We refer to this step as **"cluster assignment"**. When the assignment is complete, the algorithm proceeds to calculate new mean value of each cluster present in the data. After the recalculation of the centers, the observations are checked if they are closer to a different cluster. Using the updated cluster mean, the objects undergo reassignment. This goes on repeatedly through several iterations until the cluster assignments stop altering. The clusters that are present in the current iteration are the same as the ones obtained in the previous iteration.

Summing up the K-means clustering –

- We specify the number of clusters that we need to create.
- The algorithm selects k objects at random from the dataset. This object is the initial cluster or mean.
- The closest centroid obtains the assignment of a new observation. We base this assignment on the Euclidean Distance between object and the centroid.
- k clusters in the data points update the centroid through calculation of the new mean values present in all the data points of the cluster. The kth cluster's centroid has a length of p that contains means of all variables for observations in the k-th cluster. We denote the number of variables with p.
- Iterative minimization of the total within the sum of squares. Then through the iterative minimization of the total sum of the square, the assignment stop wavering when we achieve maximum iteration. The default value is 10 that the R software uses for the maximum iterations.

Determining Optimal Clusters

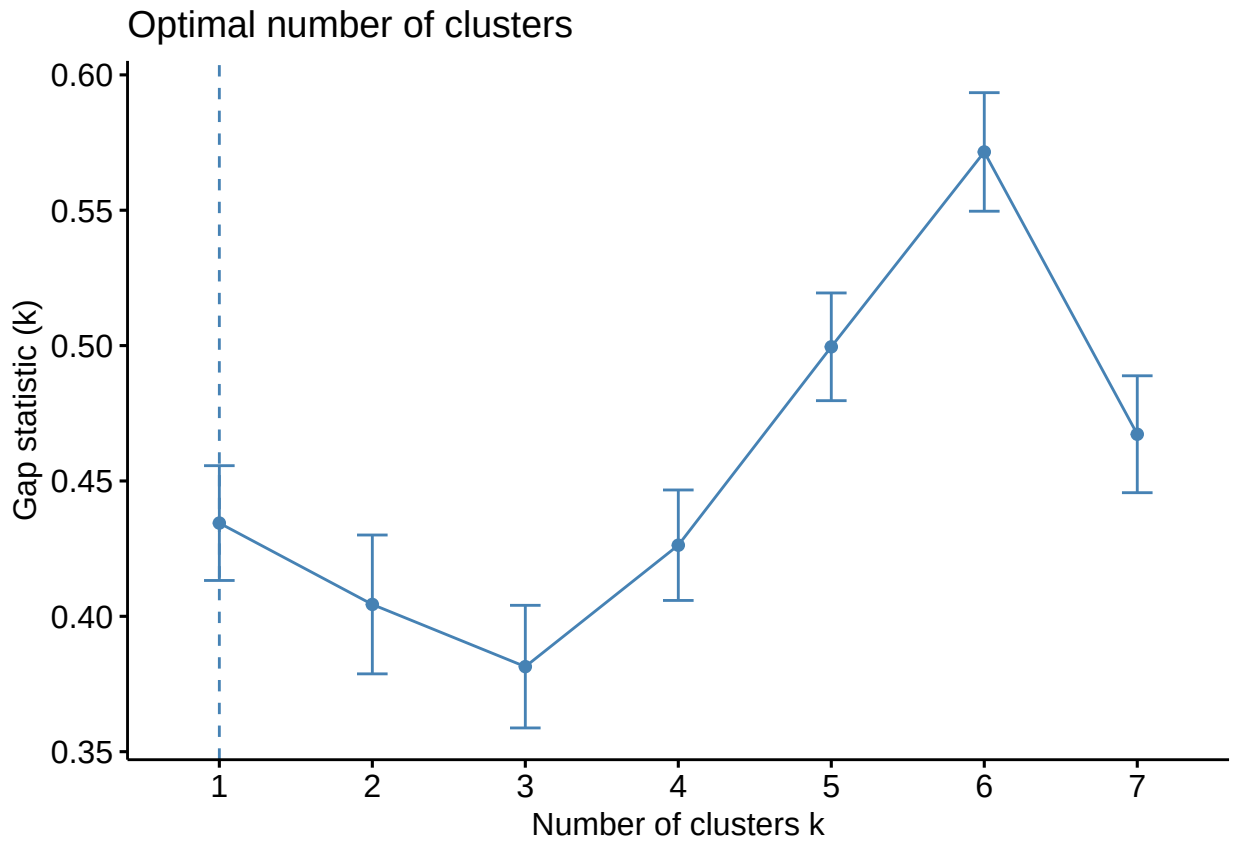
While working with clusters, you need to specify the number of clusters to use. You would like to utilize the optimal number of clusters. To help you in determining the optimal clusters, there are three popular methods –**but we're going to utilize `Gap Statistic method`**

- Elbow method
- Silhouette method
- Gap statistic

Gap Statistic Method In 2001, researchers at Stanford University – **R. Tibshirani, G.Walther and T. Hastie** published the Gap Statistic Method. We can use this method to any of the clustering method like K-means, hierarchical clustering etc. Using the gap statistic, one can compare the total intracluster variation for different values of k along with their expected values under the null reference distribution of data. With the help of **Monte Carlo simulations**, one can produce the sample dataset. For each variable in the dataset, we can calculate the range between min(xi) and max (xj) through which we can produce values uniformly from interval lower bound to upper bound.

For computing the gap statistics method we can utilize the clusGap function for providing gap statistic as well as standard error for a given output.

```
library(cluster)
set.seed(169)
nclust <- clusGap(customer_data[,3:5],
                  FUN = kmeans,
                  K.max = 7, B = 50)
library(factoextra)
fviz_gap_stat(nclust)
```




```
k6 <- kmeans(customer_data[,3:5],
             centers = 6,
             iter.max = 10,
             nstart = 1,
       algorithm = "Lloyd",
       trace = FALSE)
k6$centers
```

```
##        age annual_income_k spending_score_1_100
## 1 41.00000       109.70000             22.00000
## 2 33.36538        57.21154             49.65385
## 3 54.06000        40.46000             36.72000
## 4 25.00000        25.26087             77.60870
## 5 41.23077        79.26923             15.92308
## 6 32.69231        86.53846             82.12821
```

In the output of our kmeans operation, we observe a list with several key information. From this, we conclude the useful information being –

- **cluster** – This is a vector of several integers that denote the cluster which has an allocation of each point.
- **totss** – This represents the total sum of squares.
- **centers** – Matrix comprising of several cluster centers

- **withinss** – This is a vector representing the intra-cluster sum of squares having one component per cluster.
- **tot.withinss** – This denotes the total intra-cluster sum of squares.
- **betweenss** – This is the sum of between-cluster squares.
- **size** – The total number of points that each cluster holds.

Visualizing the Clustering Results using the First Two Principle Components

```
pcclust=prcomp(customer_data[,3:5],scale=FALSE) #principal component analysis
summary(pcclust)
```

```
## Importance of components:
##                             PC1     PC2     PC3
## Standard deviation     26.4625 26.1597 12.9317
## Proportion of Variance  0.4512  0.4410  0.1078
## Cumulative Proportion   0.4512  0.8922  1.0000
```

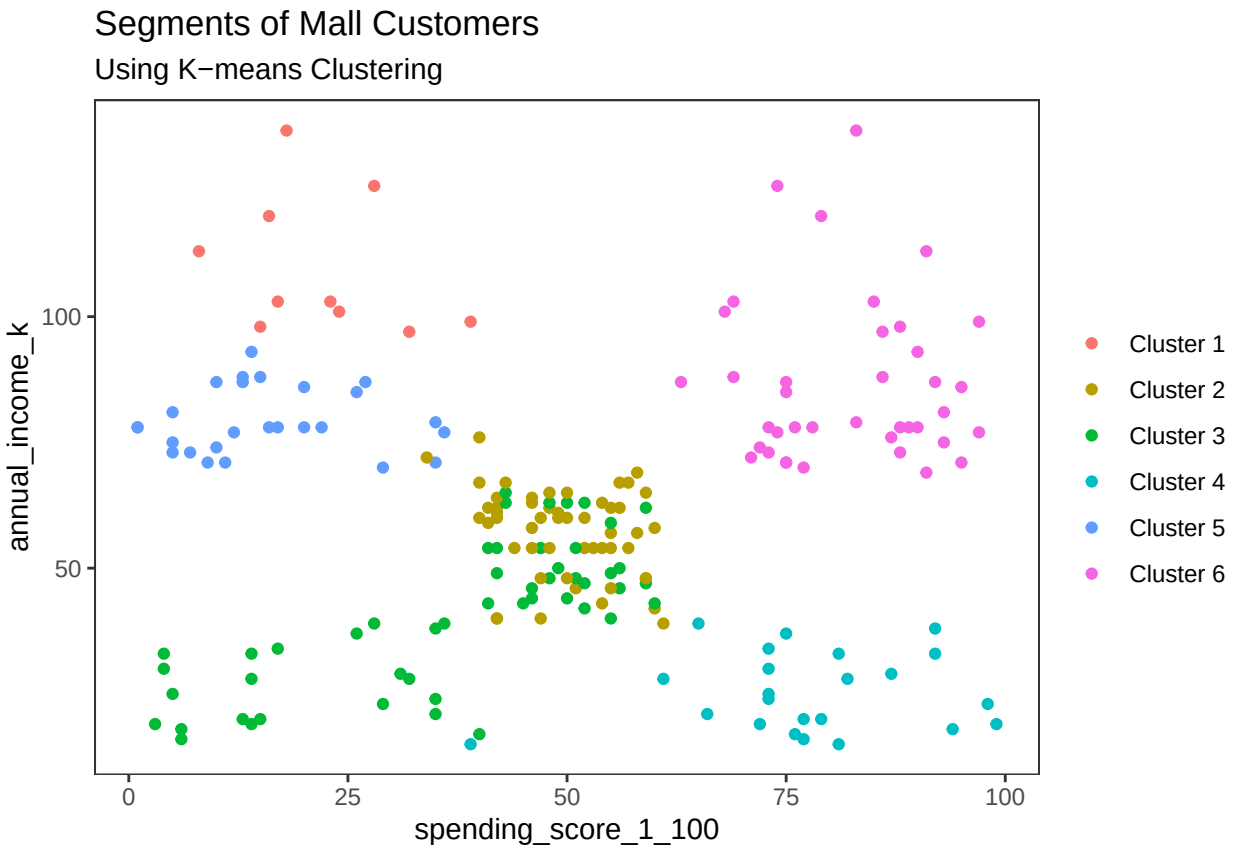
```
pcclust$rotation[,1:2]
```

```
##                              PC1        PC2
## age                    0.1889742 -0.1309652
## annual_income_k       -0.5886410 -0.8083757
## spending_score_1_100 -0.7859965  0.5739136
```

```
customer_data %>%
  ggplot(aes(x=spending_score_1_100,
             y=annual_income_k,
             )) +

  geom_point(
    stat = 'identity',
   aes(color = as.factor(k6$cluster))
  )+
  scale_color_discrete(name=" ",
              breaks=c("1", "2", "3", "4", "5","6"),
              labels=c("Cluster 1", "Cluster 2", "Cluster 3", "Cluster 4", "Cluster 5","Cluster 6")) +

  ggtitle("Segments of Mall Customers", subtitle = "Using K-means Clustering")
```



From the above visualization, we observe that there is a distribution of 6 clusters as follows –

Cluster 6 and 4 – These clusters represent the customer_data with the medium income salary as well as the medium annual spend of salary.

Cluster 1 – This cluster represents the customer_data having a high annual income as well as a high annual spend.

Cluster 3 – This cluster denotes the customer_data with low annual income as well as low yearly spend of income.

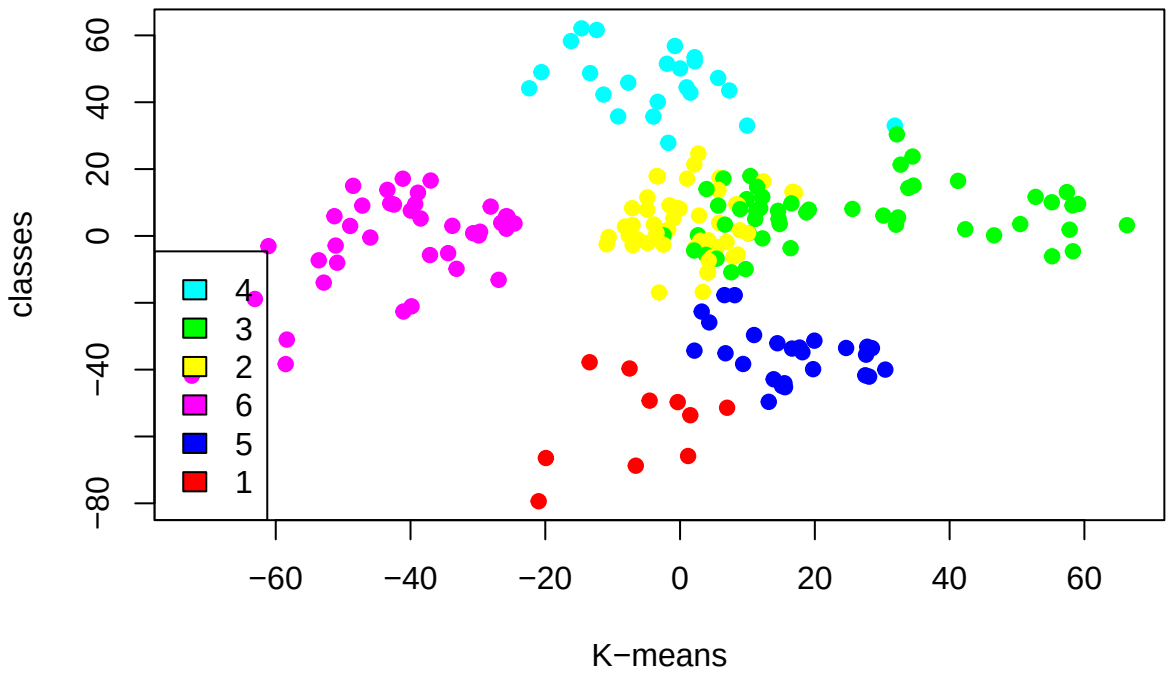
Cluster 2 – This cluster denotes a high annual income and low yearly spend.

Cluster 5 – This cluster represents a low annual income but its high yearly expenditure.

```
kCols=function(vec){cols=rainbow (length (unique (vec)))
return (cols[as.numeric(as.factor(vec))])}

digCluster<-k6$cluster; dignm<-as.character(digCluster); # K-means clusters

plot(pcclust$x[,1:2], col =kCols(digCluster),pch =19,xlab ="K-means",ylab="classes")
legend("bottomleft",unique(dignm),fill=unique(kCols(digCluster)))
```



Summary

In this data science project, we went through the customer segmentation model. We developed this using a class of machine learning known as unsupervised learning. Specifically, we made use of a clustering algorithm called K-means clustering. We analyzed and visualized the data and then proceeded to implement our algorithm. Hope you enjoyed this customer segmentation project of machine learning using R.