Project

Yuanyuan Sun, Carlos Muñoz Losa

1. Clean Environment & Load Libraries

```
# Clean environment
rm(list=ls())

# Load libraries
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.1.3
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.1.3
```

```
## corrplot 0.92 loaded
```

```
# Note! Set the new working Directory

# Every time you synchronize code from the repository, it's better to check and reset the working direc

setwd("D:/WorkSpace/RWorkSpace/DataAnalysisProjects/CustomerSegmentation")
```

2. Source, Collect and Load Data

2.1 Source and Collect Data

The goal of 'data sourcing and collection' is to find data that is relevant to solving the problem or supports an analytical solution of the stated objective.

This step involves reviewing existing data sources and finding out if it is necessary to collect new data. It may involve any number of tasks to get the data in-hand, such as querying databases, scraping data from data streams, submitting requests to other departments, or searching for third-party data sources. Here, we collect the data from third-party. Our project is about Customer Segmentation. The data set comes from Data Flair.

2.2 Load Data

```
Mall_Customers <- read.csv("Mall_Customers.csv",
                           header = TRUE,
                           sep = ",",
                           stringsAsFactors=TRUE,
                           col.names = c("CustomerID","Gender","Age","Annual Income","Spending Score"))

# Data Exploration
```

```
## Read data
head(Mall_Customers)
```

```
##   CustomerID Gender Age Annual.Income Spending.Score
## 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
```

```
## Features types
str(Mall_Customers)
```

```
## 'data.frame':    200 obs. of  5 variables:
##  $ CustomerID    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Gender        : Factor w/ 2 levels "Female","Male": 2 2 1 1 1 1 1 1 2 1 ...
##  $ Age           : int  19 21 20 23 31 22 35 23 64 30 ...
##  $ Annual.Income : int  15 15 16 16 17 17 18 18 19 19 ...
##  $ Spending.Score: int  39 81 6 77 40 76 6 94 3 72 ...
```

```
# plot(Mall_Customers)
```

In the original data set, there 200 observations, 5 features. The following work generally based on Data Analysis Process - 7 Steps.

3. Process and Clean Data

In this step, we will look for data errors, missing data, or extreme outliers etc.

3.1 Explore missing value patterns

```
# Explore missing value patterns
# Check if there are NA values

NANumbers <- sum(is.na(Mall_Customers))
paste("**** The number of NA values in this data set =", NANumbers)
```

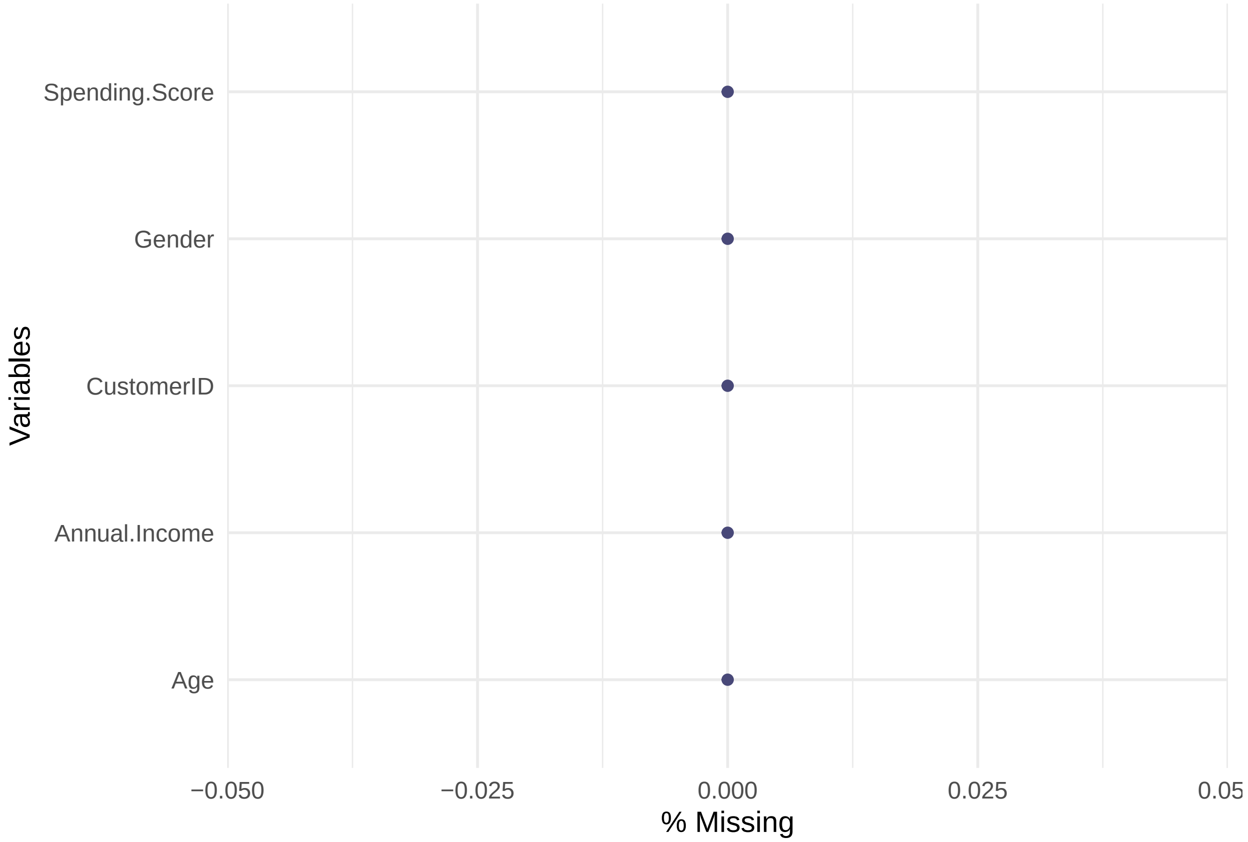
```
## [1] "**** The number of NA values in this data set = 0"
```

```
# Plotting percentage of missing values per feature
library(naniar)
```

```
## Warning: package 'naniar' was built under R version 4.1.3
```

```
gg_miss_var(Mall_Customers, show_pct = TRUE)
```

```
## Warning: The `guide` argument in `scale_*()` cannot be `FALSE`. This was deprecated in
## ggplot2 3.3.4.
## i Please use "none" instead.
## i The deprecated feature was likely used in the naniar package.
##   Please report the issue at <https://github.com/njtierney/naniar/issues>.
```



The number of NA values in this data set = 0. There are no NA values in the original data set.

3.2 Features Data Summarization

Basic statistical summary reports and charts can help reveal any serious issues or gaps in the data. In the output of summary, we can see the distribution of gender on customer data set. Apparently, the number of male customers is greater than the number of female customers. We detect that CustomerID has no meaning here. We will remove CustomerID in the following step.

```r
# Quickly summarize the values in a data frame here
summary(Mall_Customers)
```

```
##    CustomerID        Gender          Age        Annual.Income    Spending.Score
##  Min.   :  1.00   Female:112   Min.   :18.00   Min.   : 15.00   Min.   : 1.00
##  1st Qu.: 50.75   Male  : 88   1st Qu.:28.75   1st Qu.: 41.50   1st Qu.:34.75
##  Median :100.50                Median :36.00   Median : 61.50   Median :50.00
##  Mean   :100.50                Mean   :38.85   Mean   : 60.56   Mean   :50.20
##  3rd Qu.:150.25                3rd Qu.:49.00   3rd Qu.: 78.00   3rd Qu.:73.00
##  Max.   :200.00                Max.   :70.00   Max.   :137.00   Max.   :99.00
```

3.3 Data parsing (Set dummy variables)

Data parsing is converting data from one format to another. Widely used for data structuring, it is generally done to make the existing, often unstructured, unreadable data more comprehensible. We find 'Gender' is binary in the original data set. So, it's better to set the dummy variable on gender for the further data processing and analysis.

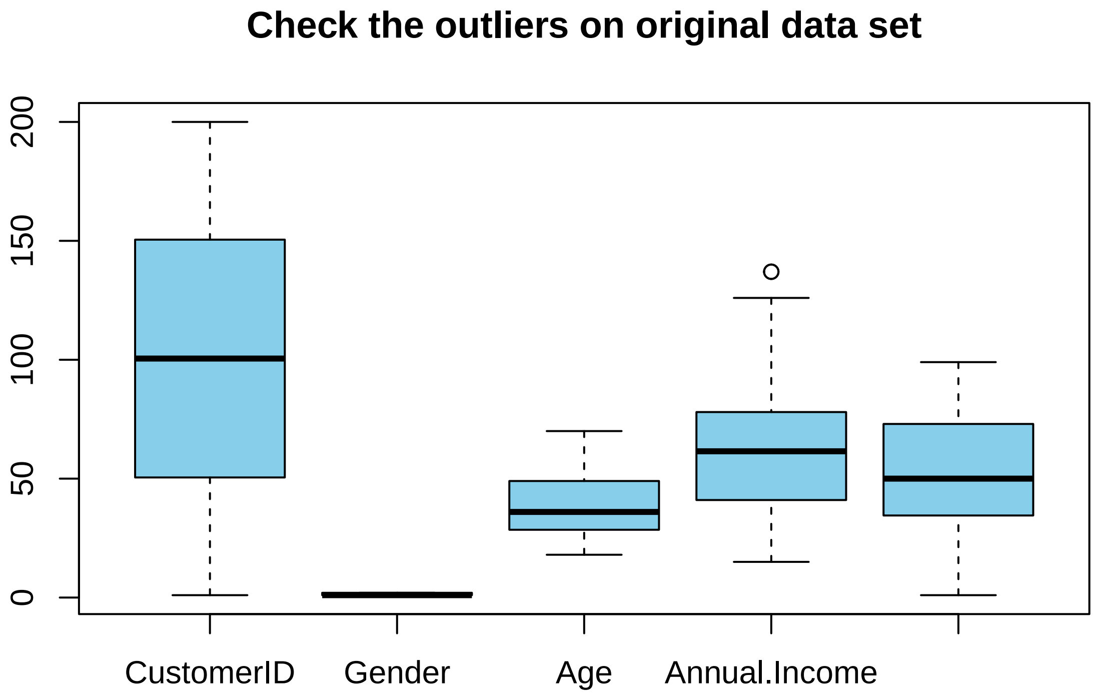
```
#  Compactly displaying the internal structure of data frame: Mall_Customers.
# str(Mall_Customers)

# Set dummy variables
Mall_Customers$Gender<-as.integer(Mall_Customers$Gender)
# Mall_Customers <- transform(Mall_Customers, Gender = as.integer(Gender))

# Display new data frame
head(Mall_Customers)
```

```
##   CustomerID Gender Age Annual.Income Spending.Score
## 1          1      2  19            15             39
## 2          2      2  21            15             81
## 3          3      1  20            16              6
## 4          4      1  23            16             77
## 5          5      1  31            17             40
## 6          6      1  22            17             76
```

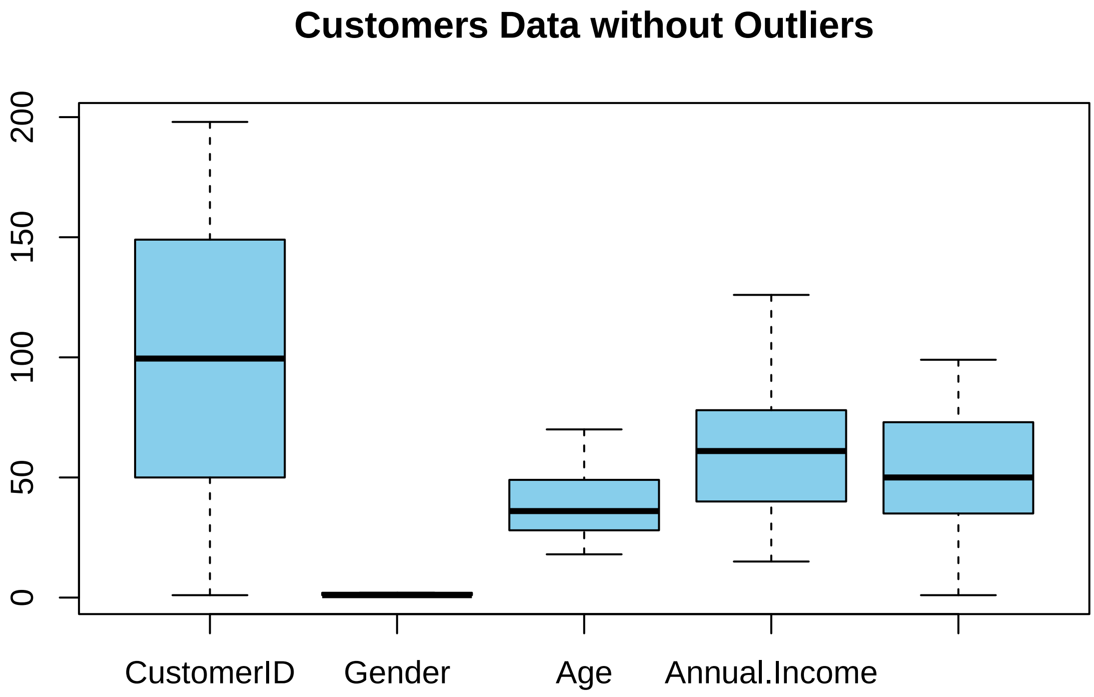
3.4 Checking & Treating Outliers

Dealing and Treating Outliers in data is always essential for accurate results. According the boxplot below of the original data set, we find the original data set has an outlier in Annual.Income feature. The outlier is out of the confidence interval. And the outlier would influence the clustering negatively. So it's better to drop that value.

```
# Handling outliers: check the outliers by the boxplotx

boxplot(Mall_Customers, col="skyblue", main="Check the outliers on original data set")
```



```
# Handling outliers: remove outliers

# print(max(Mall_Customers$Annual.Income))

Mall_Customers1 = subset(Mall_Customers, Mall_Customers$Annual.Income != max(Mall_Customers$Annual.Incom

boxplot(Mall_Customers1, col="skyblue", main="Customers Data without Outliers")
```



3.5 Data Correlation

In statistics, we're often interested in understanding the relationship between two variables. One way to quantify this relationship is to use the Pearson correlation coefficient, which is a measure of the linear association between two variables. It has a value between -1 and 1 where:

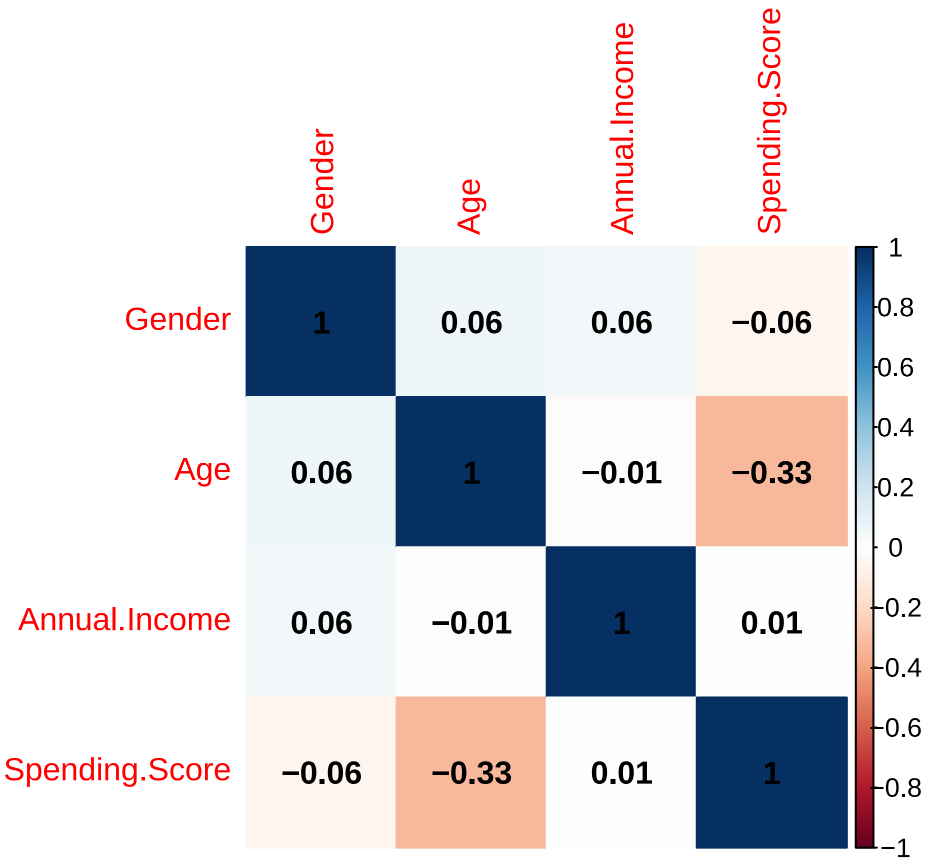
-1 indicates a perfectly negative linear correlation between two variables

0 indicates no linear correlation between two variables

1 indicates a perfectly positive linear correlation between two variables

According to the output below, we see the correlation values between two different variables are almost close to 0. The features each other are lowly correlated. It indicates no linear correlation between two different features on the original data set.

```
Mall_Customers_temp <- Mall_Customers[, -1]
corrplot(cor(Mall_Customers_temp), addCoef.col = 'black', method="color")
```



```
# TODO: analysis the correlation among features
```

4. Data Transformation

4.1 Data Transforming (Remove ID)

Data transformation is a technique used to convert the raw data into a suitable format that efficiently eases data mining and retrieves strategic information. Data transformation includes data cleaning techniques and a data reduction technique to convert the data into the appropriate form. According the detection in previous steps, we found customer ID cannot be considered as a feature. It's better to remove it.

```
# Remove ID
Mall_Customers1 <- Mall_Customers[, -1]
head(Mall_Customers1)
```

```
##   Gender Age Annual.Income Spending.Score
## 1      2  19            15             39
## 2      2  21            15             81
## 3      1  20            16              6
## 4      1  23            16             77
## 5      1  31            17             40
## 6      1  22            17             76
```

4.2 Formating Final Data Frame

According to the displaying of the internal structure of the data set. All features in the data frame are integer. There's no need to format columns.

4.3 Normalizing the Data

Since, clustering techniques use Euclidean Distance to form the cohorts, it will be wise e.g to scale the variables having heights in meters and weights in KGs before calculating the distance. As we know that feature scaling is useful in situations where a set of input features differs wildly in scale. If some of those features are thrown into a model, then the model will need to balance its scale while figuring out what to do. Drastically varying scale in input features can lead to numeric stability issues for the model training algorithm. In those situations, it's a good idea to standardize the features. Clustering algorithms such as K-means do need feature normalization before they are fed to the algo.

Reference:

Zheng, A., & Casari, A. (2018). Feature engineering for machine learning: Principles and techniques for data scientists (First edition). O'Reilly.

```
# Scale() is a generic function whose default method centers and/or scales the columns of a numeric mat
Age.Scale <- as.data.frame(scale(Mall_Customers1$Age))
Annual.Income.Scale <- as.data.frame(scale(Mall_Customers1$Annual.Income))
Spending.Score.Scale <- as.data.frame(scale(Mall_Customers1$Spending.Score))
# Annual.Income.Scale

df.customers <- data.frame(Gender = Mall_Customers$Gender,
                           Age = Age.Scale,
                           Annual.Income = Annual.Income.Scale,
                           Spending.Score = Spending.Score.Scale)
colnames(df.customers)[2] <- "Age"
colnames(df.customers)[3] <- "Annual.Income"
colnames(df.customers)[4] <- "Spending.Score"
head(df.customers)
```

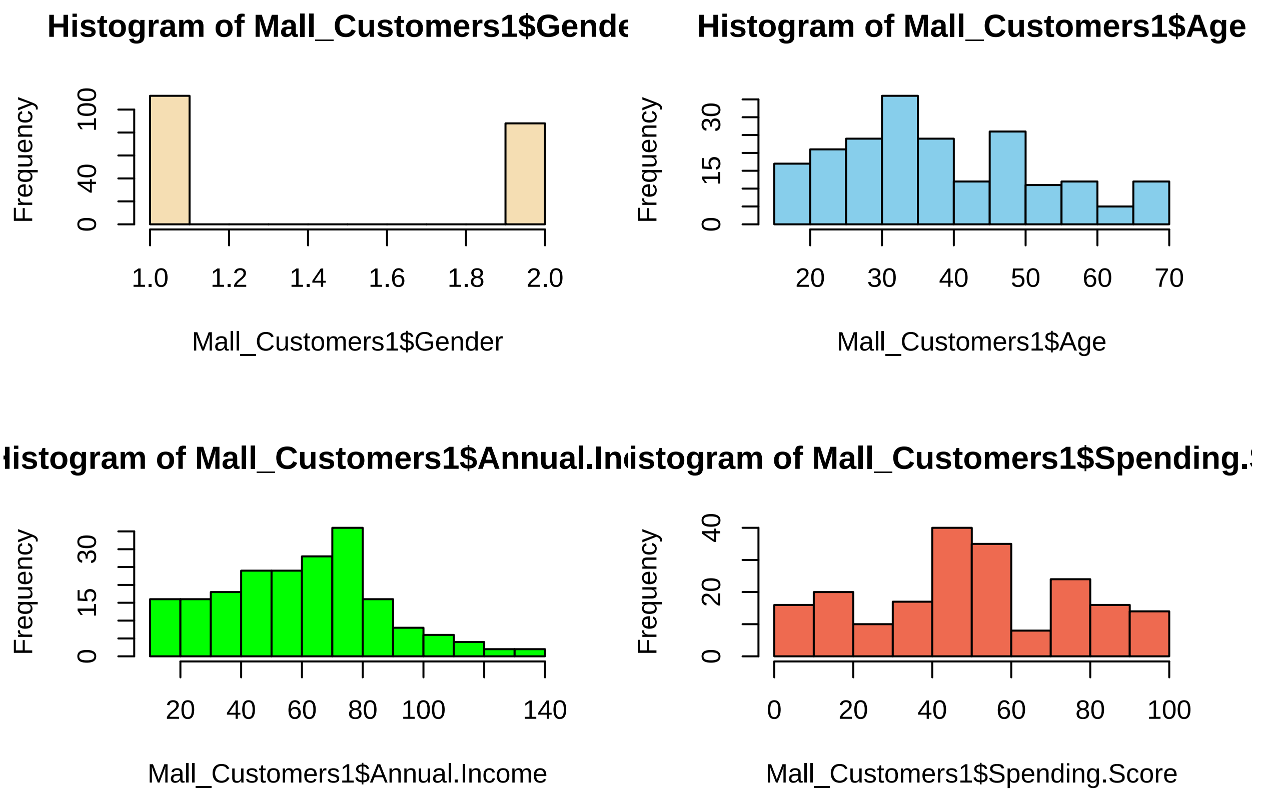
```
##   Gender        Age Annual.Income Spending.Score
## 1      2 -1.4210029     -1.734646     -0.4337131
## 2      2 -1.2778288     -1.734646      1.1927111
## 3      1 -1.3494159     -1.696572     -1.7116178
## 4      1 -1.1346547     -1.696572      1.0378135
## 5      1 -0.5619583     -1.658498     -0.3949887
## 6      1 -1.2062418     -1.658498      0.9990891
```

4.3 Univariate Distribution

4.3.1 Histogram

The peaks represent the most common values. The most common values on 'age' feature are 30. The most common values on 'Spending.Score' feature are around 50. We will do data analysis in detail on data visualization part later.

```
par(mfrow=c(2,2))
hist(Mall_Customers1$Gender, col = "wheat")
hist(Mall_Customers1$Age, col = "skyblue")
hist(Mall_Customers1$Annual.Income, col = "green")
hist(Mall_Customers1$Spending.Score, col = "coral2")
```



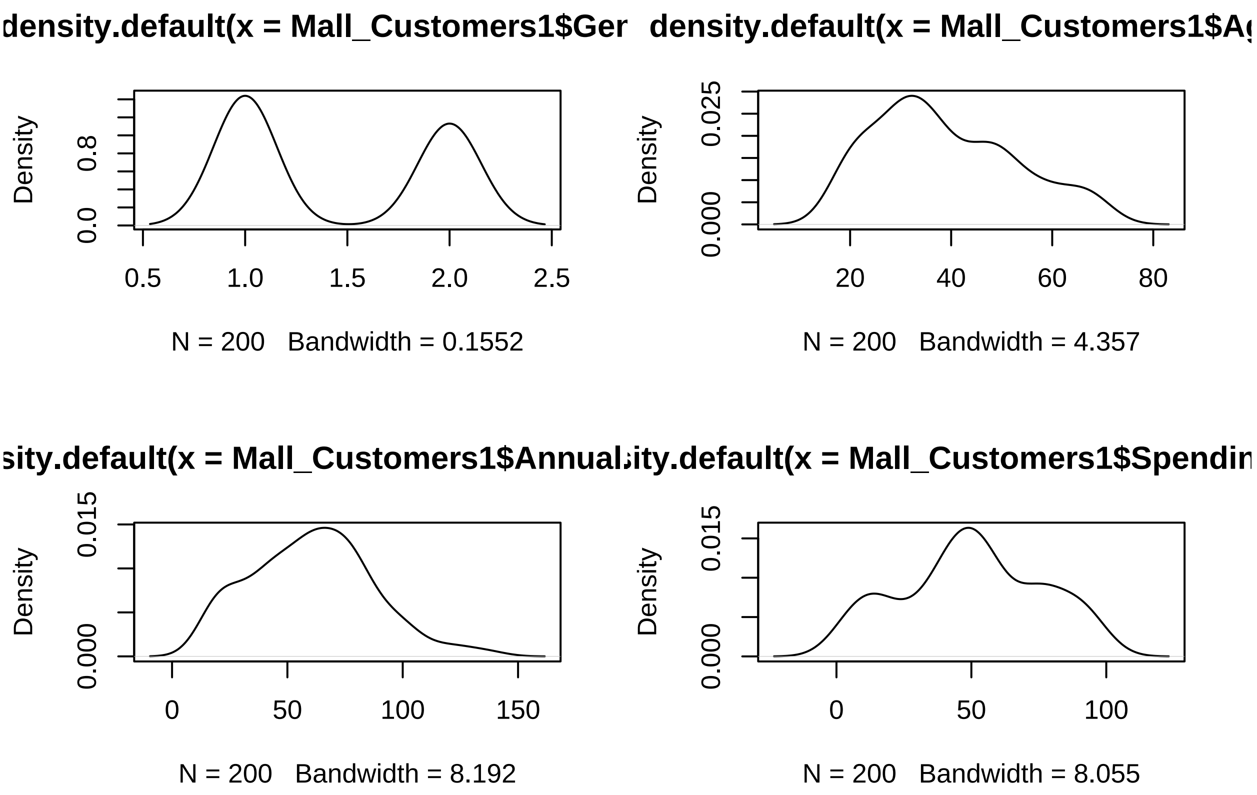
4.3.2 Kernel Density Plot

Density curves allow us to quickly see whether or not a graph is left skewed, right skewed, or has no skew. We see there are no features that their density have no skew. Except for gender, the density plot of features which are ‘Annual.Income’, ‘Spending.Score’ are right skewed. It means the mean is greater than the median on features which are ‘Annual.Income’, ‘Spending.Score’.

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

```
# Ref: https://www.statology.org/density-curves/
# Ref: https://www.r-bloggers.com/2021/11/how-to-perform-univariate-analysis-in-r/

par(mfrow=c(2,2))

plot(density(Mall_Customers1$Gender))
plot(density(Mall_Customers1$Age))
plot(density(Mall_Customers1$Annual.Income))
plot(density(Mall_Customers1$Spending.Score))
```



4.3.3 Kernel Q-Q Plot

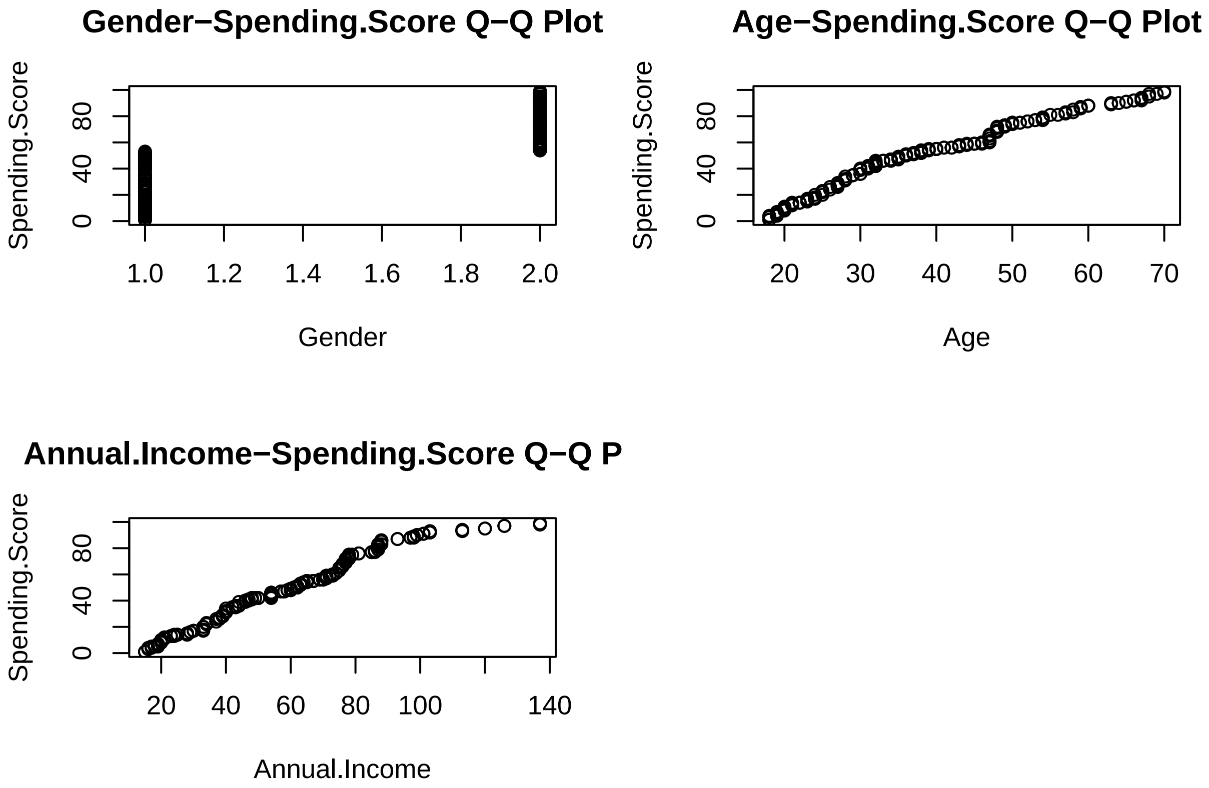
As we know that Q-Q Plot is just a visual check, not an air-tight proof, it is somewhat subjective. But it allows us to see at-a-glance if our assumption is plausible, and if not, how the assumption is violated and what data points contribute to the violation.

According to the output of the Q-Q Plot below, we see on (Spending.Score, age), (Spending.Score, Annual.Income) the points forming lines that are roughly straight. On (Spending.Score, gender), we clearly see the distribution is not normal. Non-normal distributions may lack symmetry, may have extreme values, or may have a flatter or steeper "dome" than a typical bell. There is nothing inherently wrong with non-normal data; some traits simply do not follow a bell curve. For example, data about coffee and alcohol consumption are rarely bell shaped. Since we clearly see it's not a normal distribution. So, it's better to remove the feature 'Gender' and do an experiment on model fitting to see what would happen and compare it with the other models.

The topic here we would be interested in customer segmentation is: are there any relationships among features? Based on empirical experience, people who have higher Annual.Income usually have higher Spending.Score. If we have time to figure out something among features, we would like to do linear regression on these features to see the relationships on (Spending.Score, age), (Spending.Score, Annual.Income), and compare it with the empirical distribution and normal distribution.

```
# Ref: https://towardsdatascience.com/q-q-plots-explained-5aa8495426c0
par(mfrow=c(2,2))

ySpendingScore <- Mall_Customers1$Spending.Score

# 01 Gender-Spending.Score Q-Q Plot
```

```
xGender <- Mall_Customers1$Gender

qqplot(xGender, ySpendingScore, xlab = "Gender", ylab = "Spending.Score", main = "Gender-Spending.Score
# qqline(ySpendingScore, col = "steelblue", lwd = 2)

# 02 Age-Spending.Score Q-Q Plot
xAge <- Mall_Customers1$Age

qqplot(xAge, ySpendingScore, xlab = "Age", ylab = "Spending.Score", main = "Age-Spending.Score Q-Q Plot

# 03 Annual.Income-Spending.Score Q-Q Plot
xAnnualIncome <- Mall_Customers1$Annual.Income

qqplot(xAnnualIncome, ySpendingScore, xlab = "Annual.Income", ylab = "Spending.Score", main = "Annual.I
```



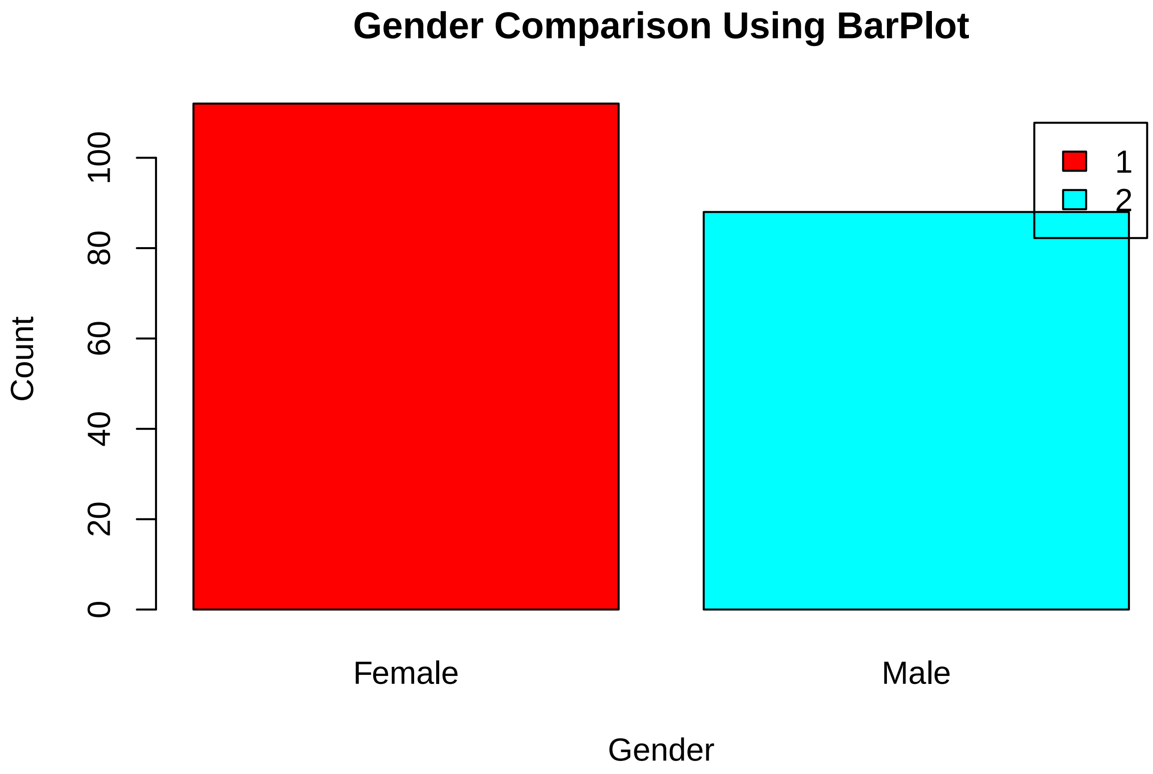
5. Visualization and Analysis of Features

5.1 Visualization of Gender

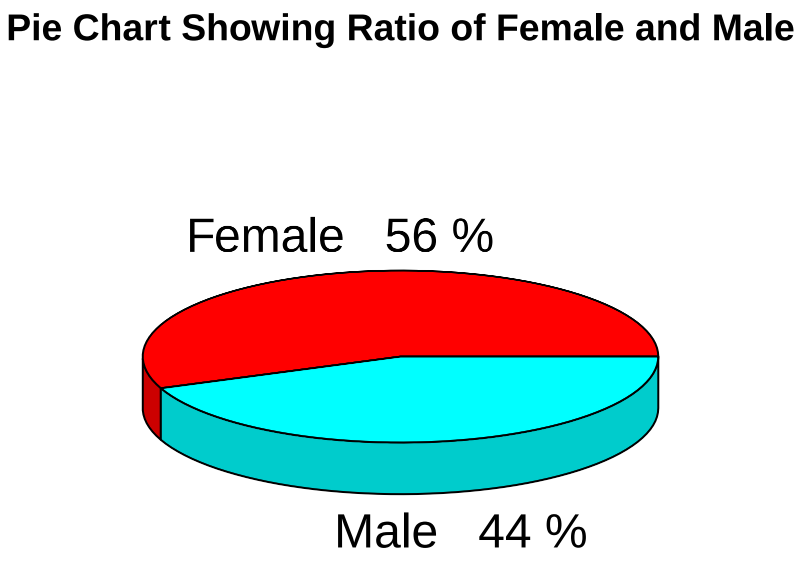
We can see clearly that Population of Female is more than Male. We can also see in Pie Chart that Female is 56% and Male is 44% in whole Population.

According the output of Kernel Q-Q Plot, ‘Gender’ has no relationship with Spending.Score. For marketing analysis and insights, we would like to see and compare the population of Female and Male.

```
genderTable <- table(Mall_Customers1$Gender)

# barplot
barplot(genderTable, main = "Gender Comparison Using BarPlot",
       ylab = "Count",
       xlab = "Gender",
       col = rainbow(2),
       legend = rownames(genderTable),
       names.arg = c("Female", "Male"))
```



```
# A pie chart
pct <- round(genderTable/sum(genderTable)*100)
lbs <- paste(c("Female","Male"), " ", pct, "%", sep=" ")

library(plotrix)

pie3D(genderTable, labels=lbs, main="Pie Chart Showing Ratio of Female and Male")
```



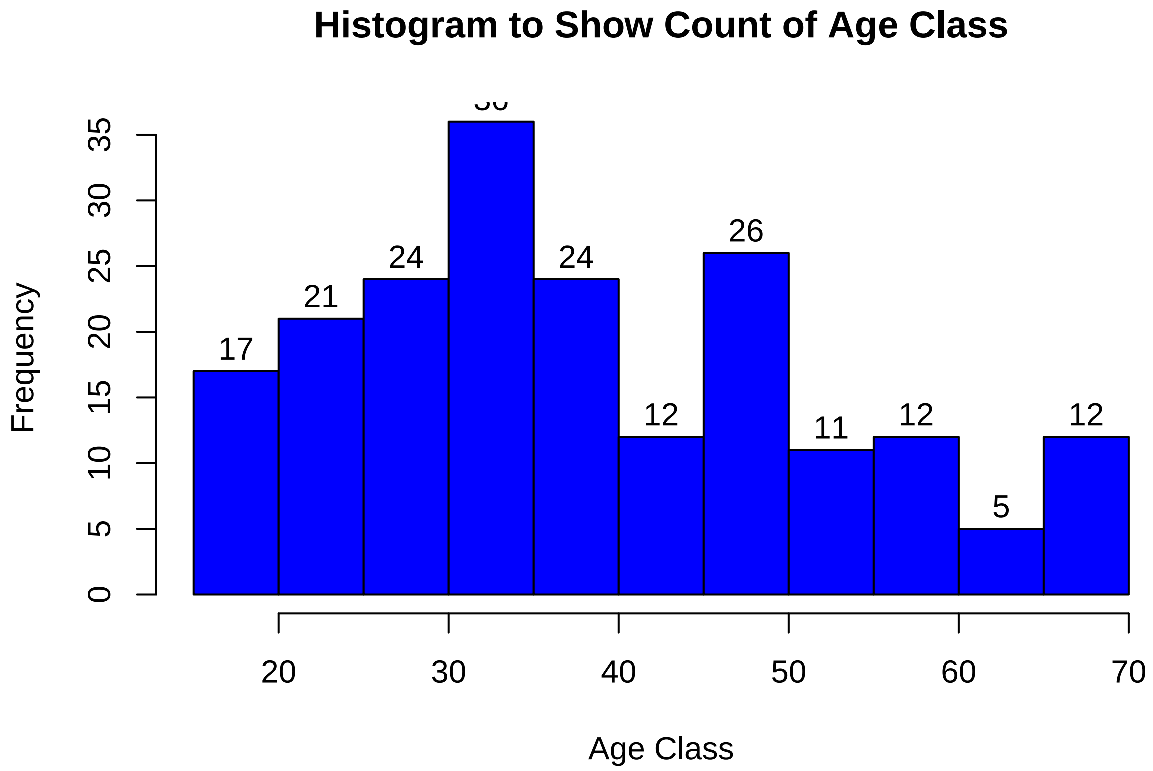
5.2 Visualization/Analysis of Age

We can see that the maximum population is between 30 to 35 on age group . We can alse see Descreptive Analysis that Minimum age is 18, Maximum age is 70 and avg. age is 38.85.

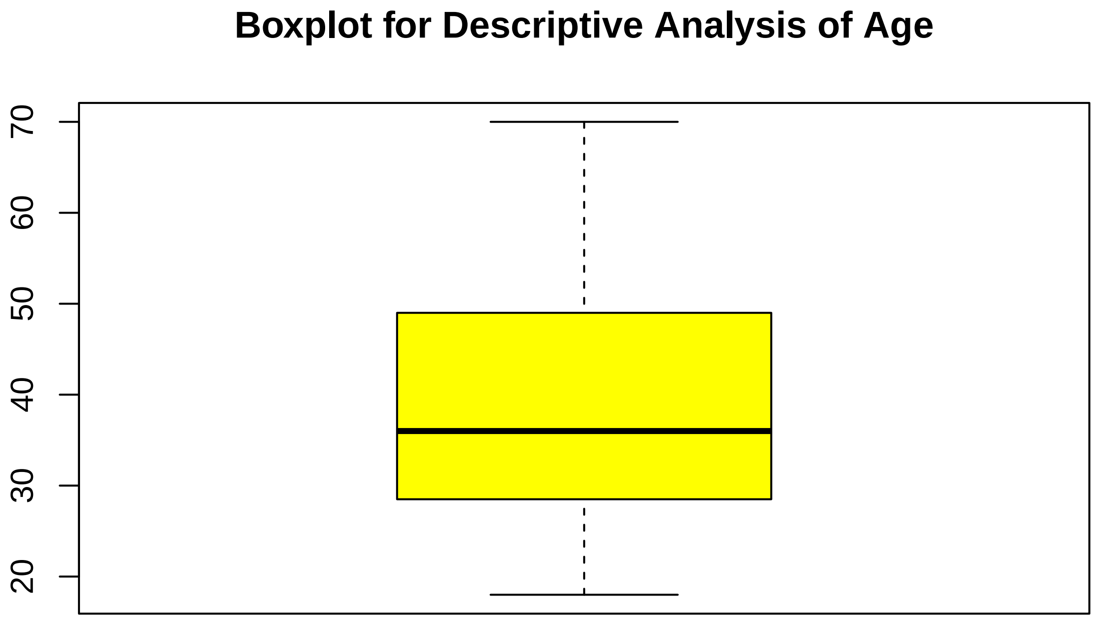
```
summary(Mall_Customers1$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   28.75   36.00   38.85   49.00   70.00
```

```
# hist plot
hist(Mall_Customers1$Age,
     col="blue",
     main="Histogram to Show Count of Age Class",
     xlab="Age Class",
     ylab="Frequency",
     labels=TRUE)
```



```
# boxplot plot
boxplot(Mall_Customers1$Age, col="yellow", main="Boxplot for Descriptive Analysis of Age")
```



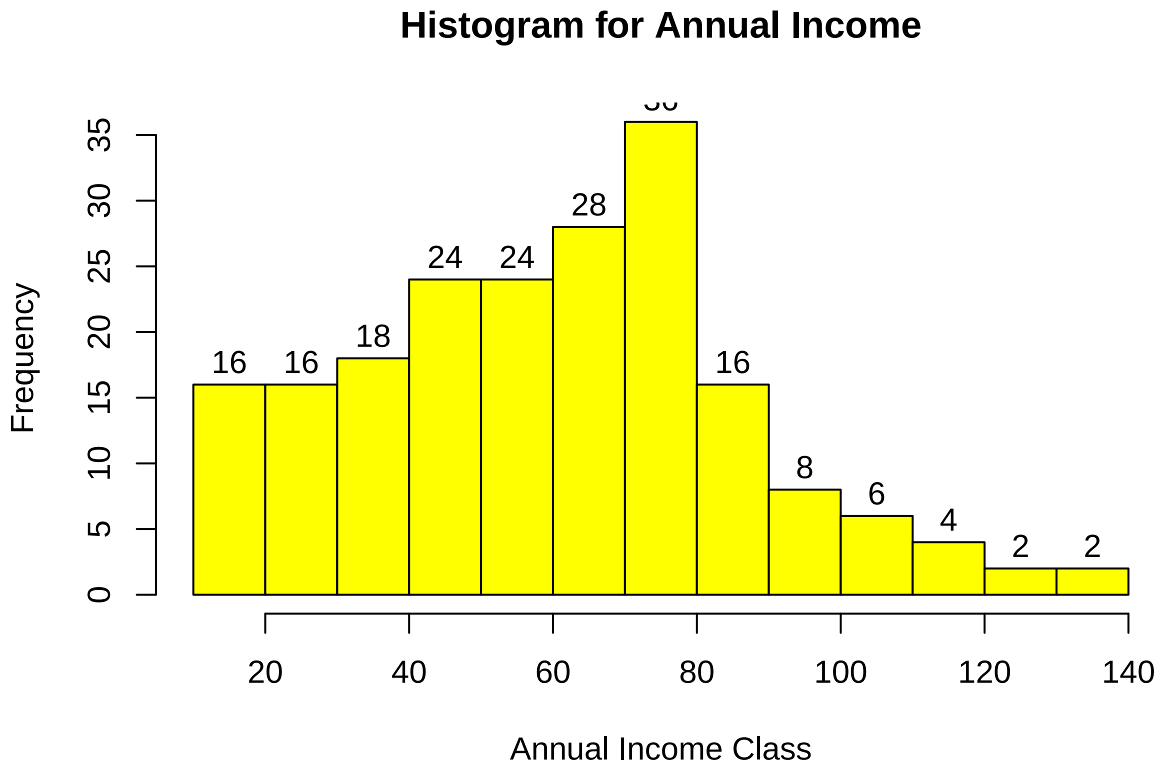
5.3 Analysis of Annual Income

We can see Descriptive Analysis that Minimum Annual Income is 15 and Maximimum is 137 with an avg. annual income of 60.56 unit. In the histogram, that Maximum Population have Annual Income between 70 to 80 units. We can see in Kernel Density Plot of Annual Income, Annual Income is distributed almost normally.

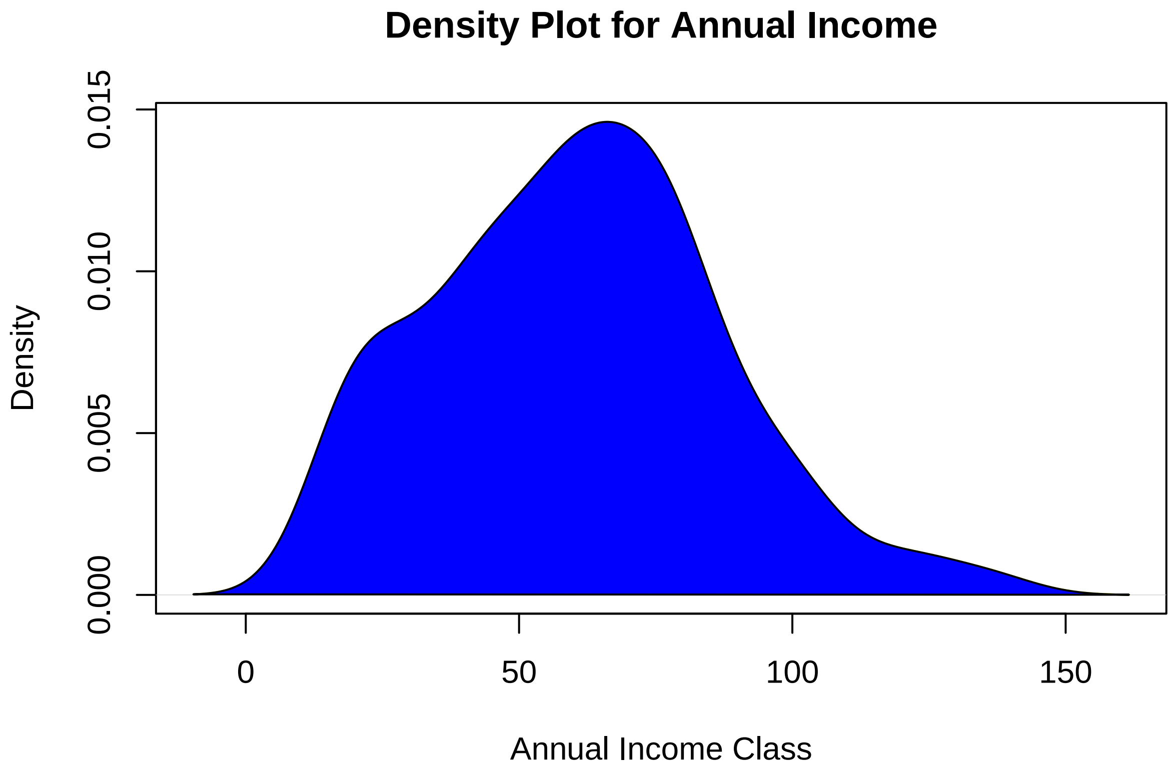
```
summary(Mall_Customers1$Annual.Income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.00   41.50   61.50   60.56   78.00  137.00
```

```
hist(Mall_Customers1$Annual.Income,
     col="yellow",
     main="Histogram for Annual Income",
     xlab="Annual Income Class",
     ylab="Frequency",
     labels=TRUE)
```



```
plot(density(Mall_Customers1$Annual.Income),
     col="yellow",
     main="Density Plot for Annual Income",
     xlab="Annual Income Class",
     ylab="Density")


polygon(density(Mall_Customers1$Annual.Income),
        col="blue")
```



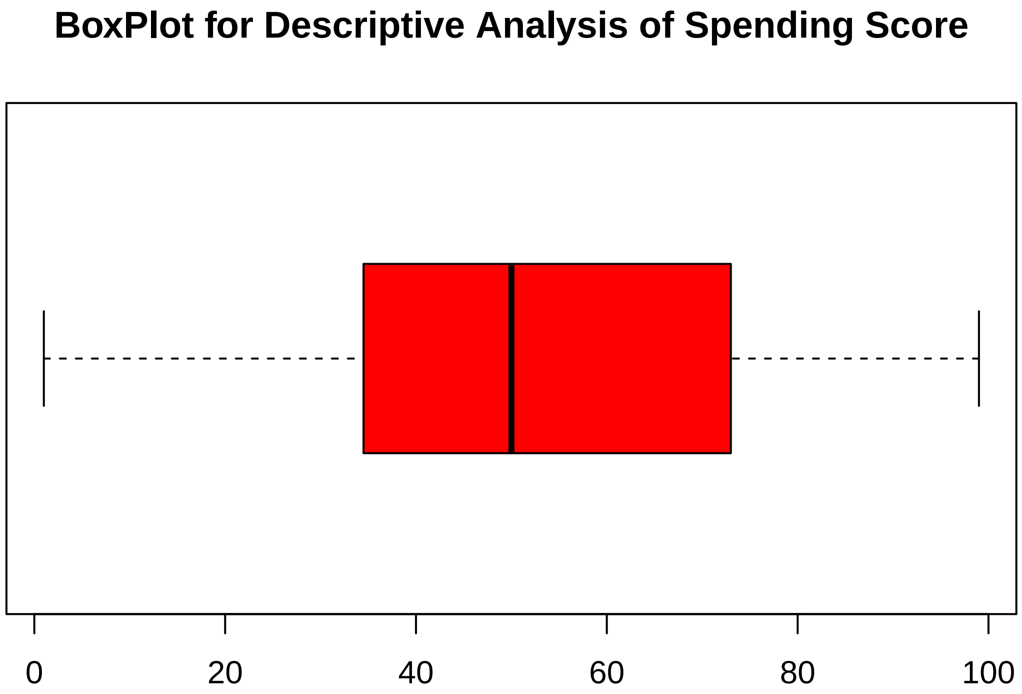
5.4 Analysis of Spending Score

We can see Descriptive Analysis of Spending Score is that Min is 1, Max is 99 and avg. is 50.20. We can visualize Descriptive Analysis with BoxPlot. In Histogram that most people have Spending Score between 40 and 50.

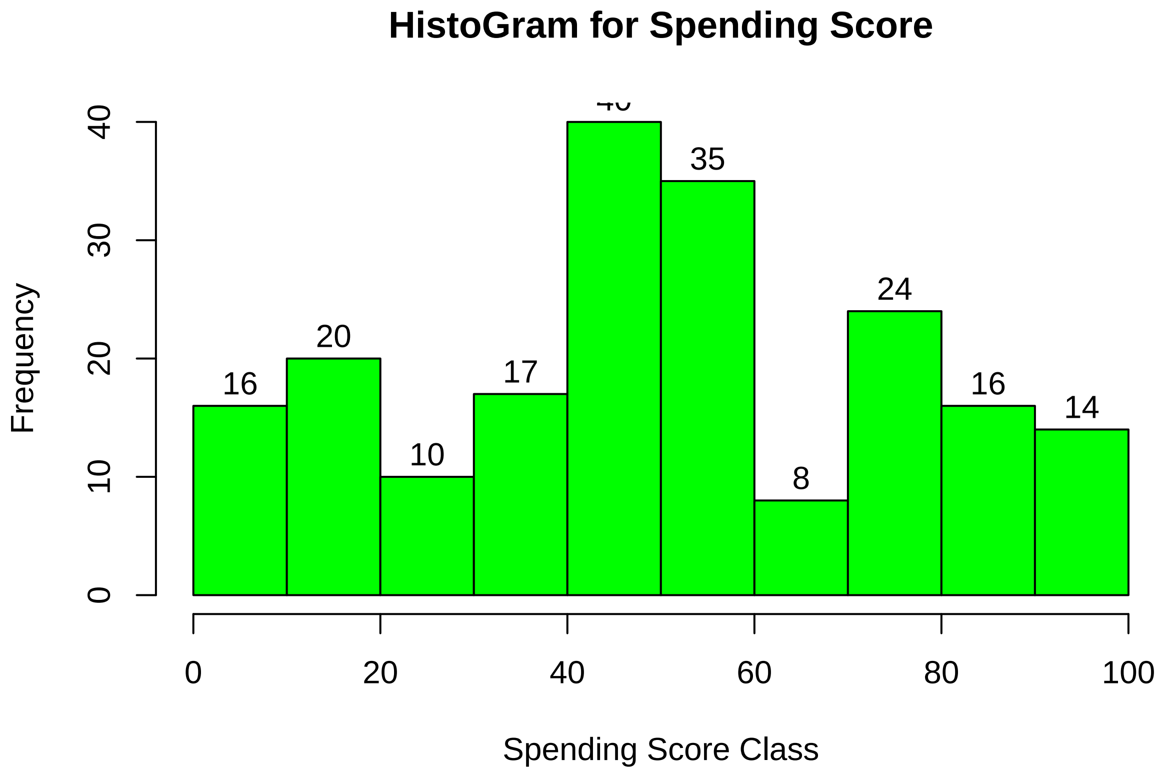
```
summary(Mall_Customers1$Spending.Score)
```

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

```
boxplot(Mall_Customers1$Spending.Score,
        horizontal=TRUE,
        col="red",
        main="BoxPlot for Descriptive Analysis of Spending Score")
```



```
hist(Mall_Customers1$Spending.Score,
     main="HistoGram for Spending Score",
     xlab="Spending Score Class",
     ylab="Frequency",
     col="green",
     labels=TRUE)
```



6. Data Analysis Stage

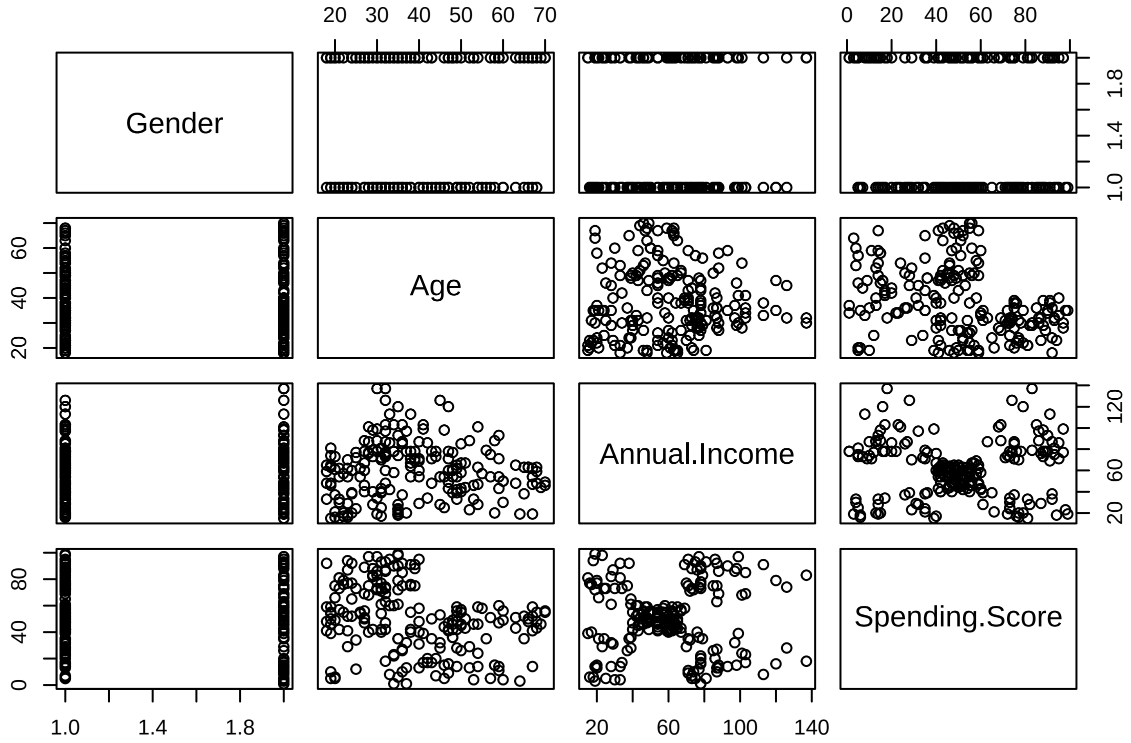
6.1 Initial Exploration of Data Clusters

Before we split the data, let's see the clusters in the original data set on the better features we choose. visually, we see there are 7 clusters on the plot below.

```
paste("Here are clusters on the original data set")
```

```
## [1] "Here are clusters on the original data set"
```

```
plot(Mall_Customers1)
```



6.3 Choose the Optimal Number of Clusters

6.3.1 the experiment on elbow method

Using the “wss” method, draw the scree plot for the optimal number of clusters.

```
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 4.1.3
```

```
library(factoextra)
```

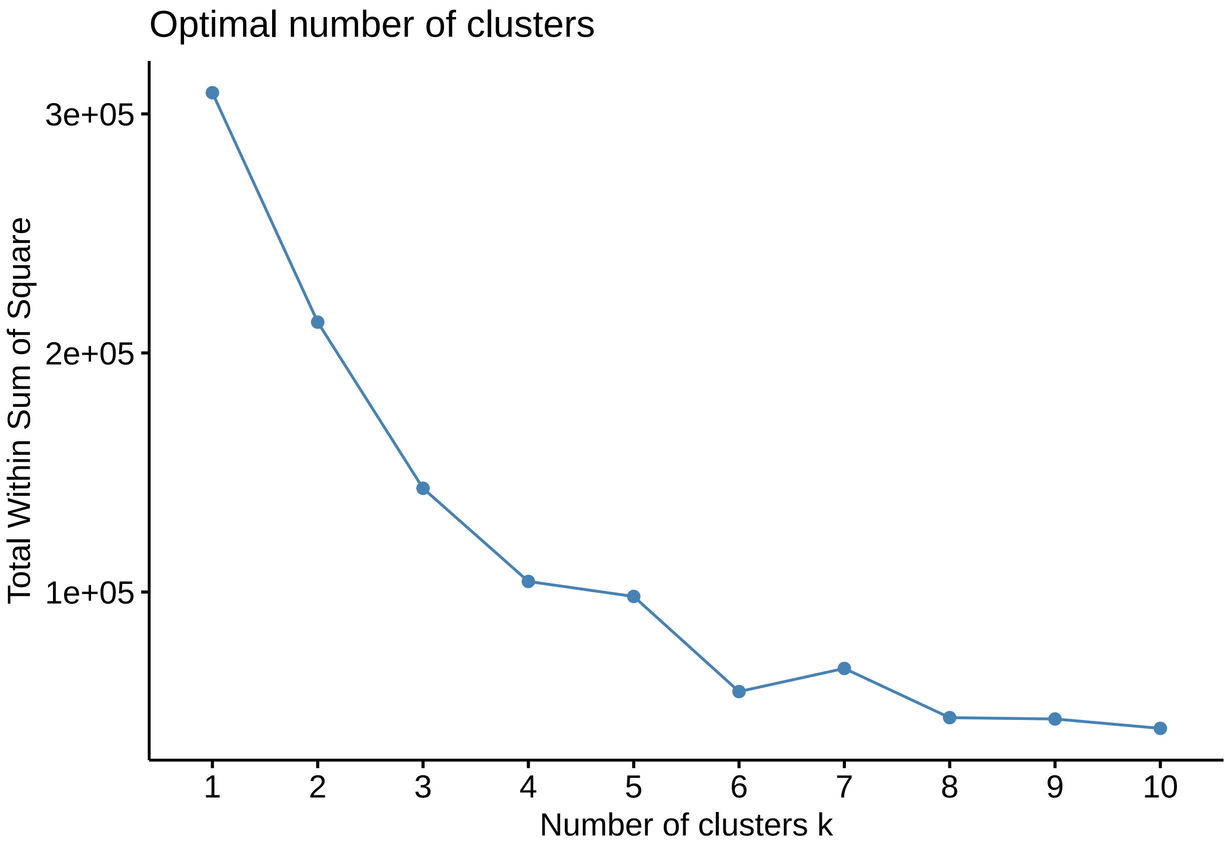
```
## Warning: package 'factoextra' was built under R version 4.1.3
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
# library(dbscan)
library(fpc)
```

```
## Warning: package 'fpc' was built under R version 4.1.3
```

```
fviz_nbclust(Mall_Customers1, kmeans, method="wss")
```



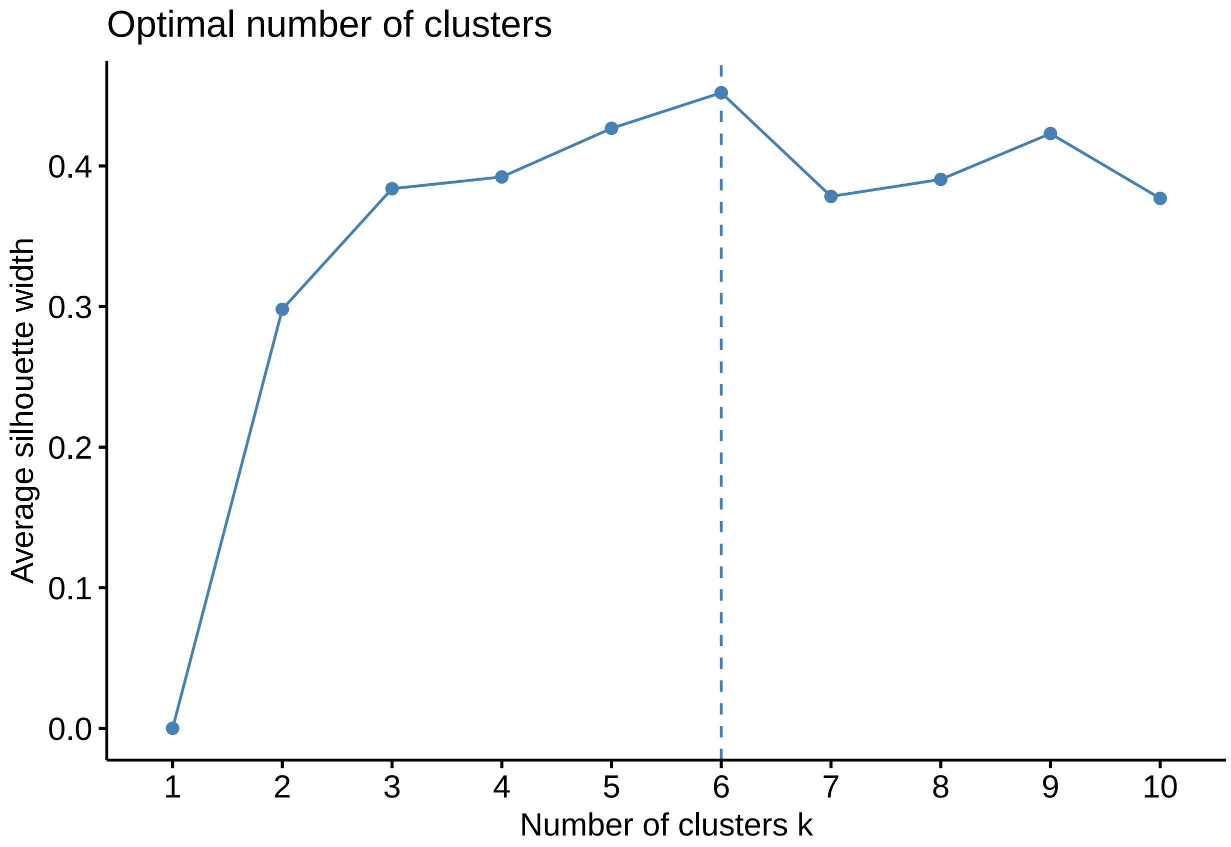
We use the “silhouette” method to draw the scree plot for the optimal number of clusters.

What do you think is the appropriate number of clusters based on elbow on this dataset?

The location of a bend or a knee is the indication of the optimum number of clusters on elbow method. We see the location of a bend could be k=3 or k=5 or k =6. However, the elbow method is somewhat subjective, different people may identify the elbow at different locations. Some may argue that k=3 or k=5 is the elbow, some may say K=6 is the elbow in the “wss” method plot. Moreover, the elbow may not be always apparent.

6.3.2 the experiment on silhouette method

```
fviz_nbclust(Mall_Customers1, kmeans, method="silhouette")
```



What do you think is the appropriate number of clusters based on silhouette on this dataset?

In the "silhouette" method plot, it seems there are two peaks. One of the silhouette coefficient peaks at K=3, and the other is at K=6. It's a bit hard to determine the optimum K. The silhouette coefficient may provide a more objective means to determine the optimal number of clusters. It performs K-Means clustering over a range of k, finds the optimal K that produces the largest silhouette coefficient, and assigns data points to clusters based on the optimized K. But when there are more than one peaks, it's a bit hard to choose the optimal K.

Experiment 1: We choose k=5 as the best number of clusters.

Experiment 2: We choose k=3 as the best number of clusters.

Experiment 3: We choose k=6 as the best number of clusters.

6.4 Model Fitting & Model Accuracy

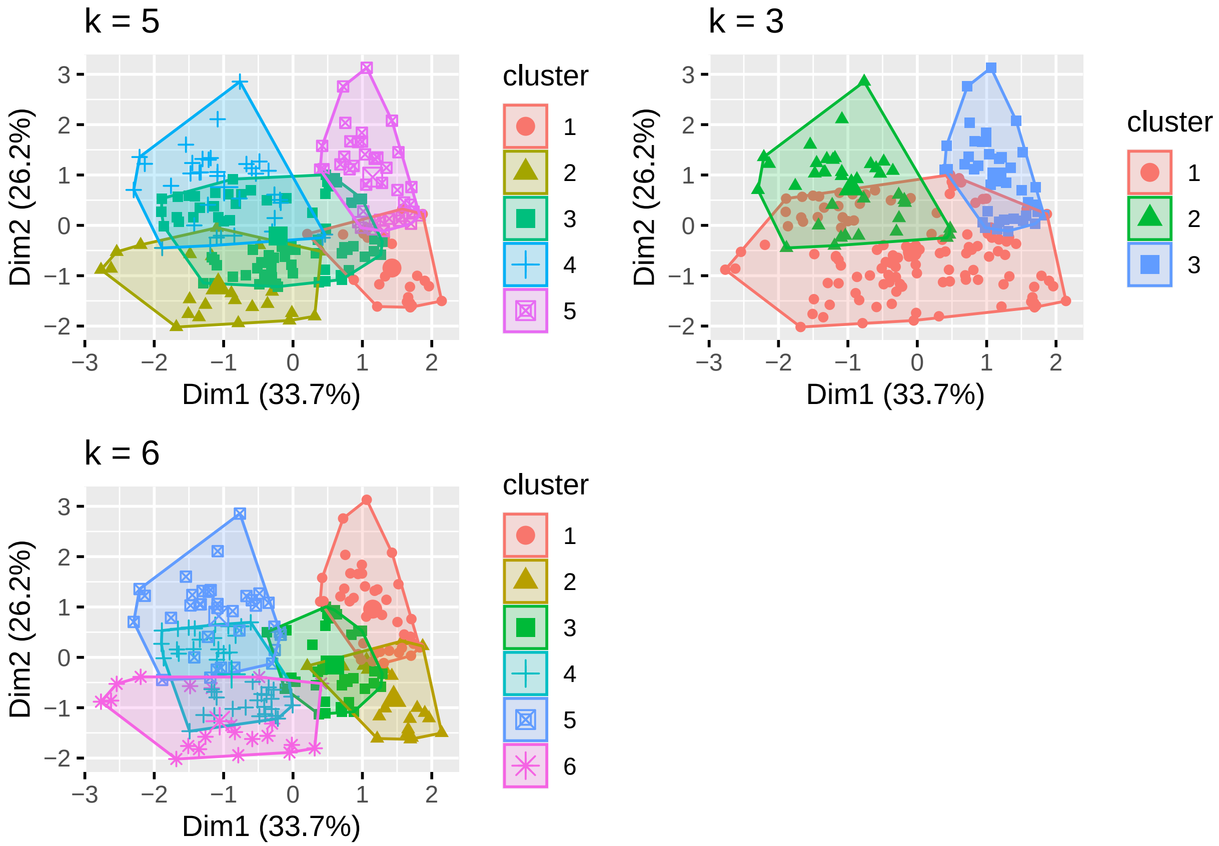
6.4.1 Model Fitting on Experiments of the Optimal Clusters

```
k5 <- kmeans(Mall_Customers1, centers = 5, nstart = 25)
k3 <- kmeans(Mall_Customers1, centers = 3, nstart = 25)
k6 <- kmeans(Mall_Customers1, centers = 6, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k5, geom = "point", data = Mall_Customers1) + ggtitle("k = 5")
p2 <- fviz_cluster(k3, geom = "point",  data = Mall_Customers1) + ggtitle("k = 3")
p3 <- fviz_cluster(k6, geom = "point",  data = Mall_Customers1) + ggtitle("k = 6")
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.1.3
```

```
grid.arrange(p1, p2, p3, nrow = 2)
```



```
cat(sprintf("1. k = 5, the Between SS / Total SS is %.3f", k5$betweenss/k5$totss), "\n")
```

```
## 1. k = 5, the Between SS / Total SS is 0.756
```

```
cat(sprintf("2. k = 3, the Between SS / Total SS is %.3f", k3$betweenss/k3$totss), "\n")
```

```
## 2. k = 3, the Between SS / Total SS is 0.536
```

```
cat(sprintf("3. k = 6, the Between SS / Total SS is %.3f", k6$betweenss/k6$totss), "\n")
```

```
## 3. k = 6, the Between SS / Total SS is 0.811
```

1. k = 5, the Between SS / Total SS is 0.860
2. k = 3, the Between SS / Total SS is 0.732
3. k = 6, the Between SS / Total SS is 0.885

When k=5, the Between SS / Total SS is 0.860. As we learned that Between SS / Total SS is between 0.0 and 1.0. The higher this ratio, the more variance is explained by the clusters. Between SS / Total SS is 0.860 that indicates a good fit.

When k=3, We see the value on "Between SS / Total SS" is much smaller than the value we calculated on experiment 1; Hence, k = 3 is not the optimal number of clusters.

When k=6, We see the value on "Between SS / Total SS" is greater than the value we calculated on experiment

1; Hence, k = 6 is the optimal number of clusters.

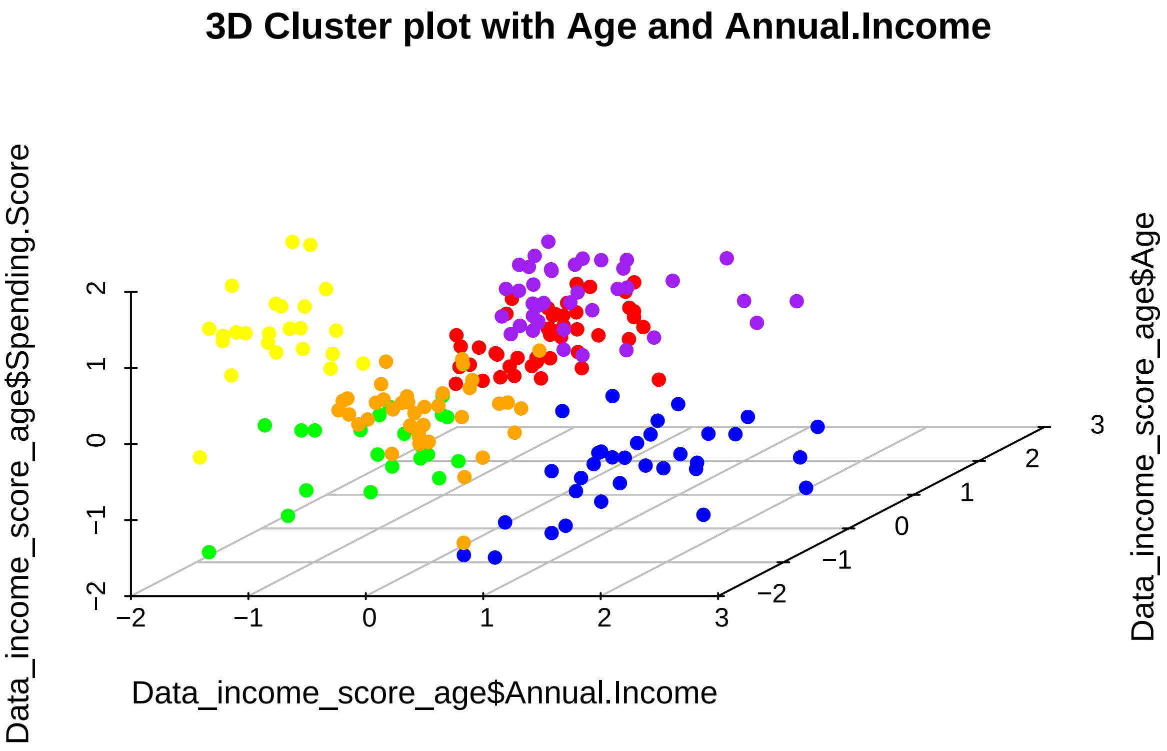
By running K-means clustering three times using 3 different initial, k = 6 is the optimal number of clusters.

6.4.2 Model Fitting on the Optimal Number of Cluster

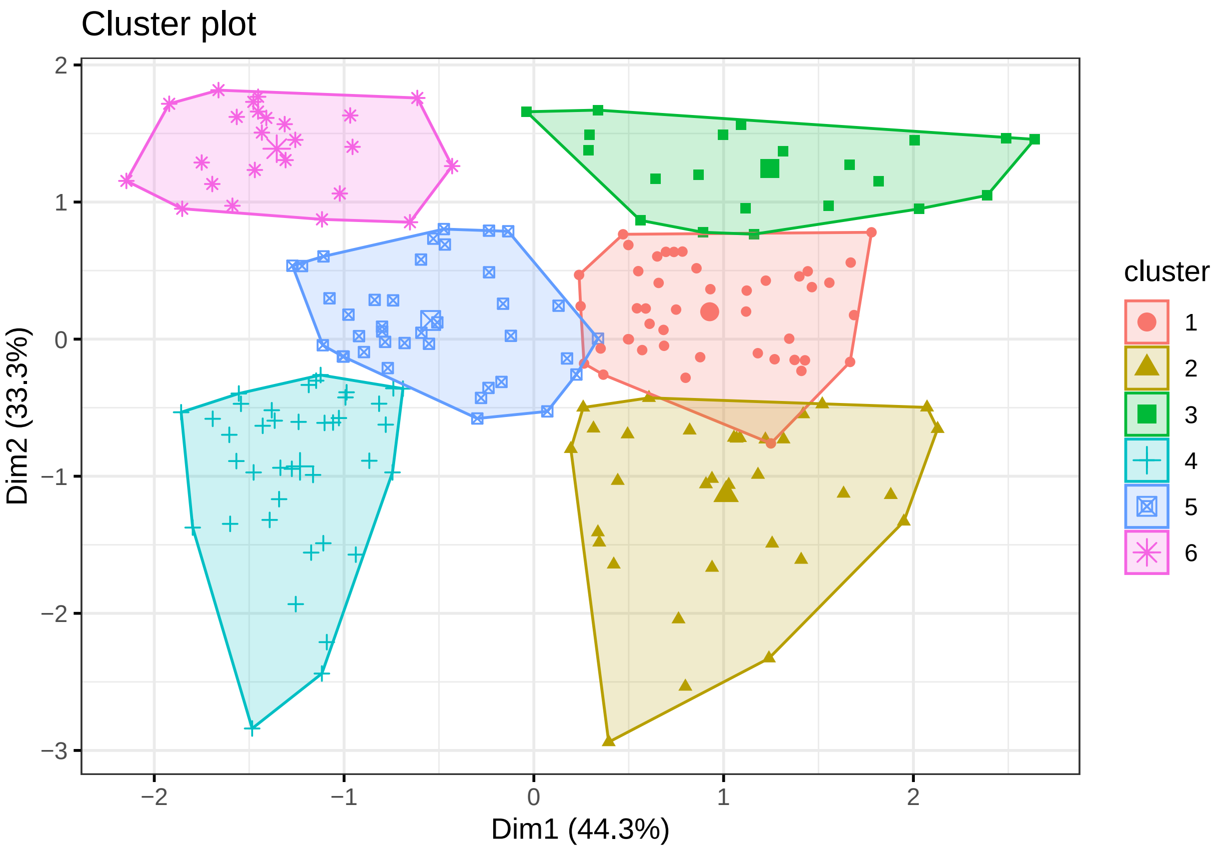
```
# Reduce gender
Data_income_score_age <- df.customers[-1]

kmeansModel_income_score_age = kmeans(Data_income_score_age,6, nstart = 25)

library(scatterplot3d)
```

```
## Warning: package 'scatterplot3d' was built under R version 4.1.3
```

```
colors <- c("red", "blue", "green", "purple", "orange", "yellow")
colors <- colors[kmeansModel_income_score_age$cluster]

scatterplot3d(x = Data_income_score_age$Annual.Income,
              y = Data_income_score_age$Age,
              z = Data_income_score_age$Spending.Score,
              box = FALSE,
              color = colors,
              pch=16,
              main = "3D Cluster plot with Age and Annual.Income")
```



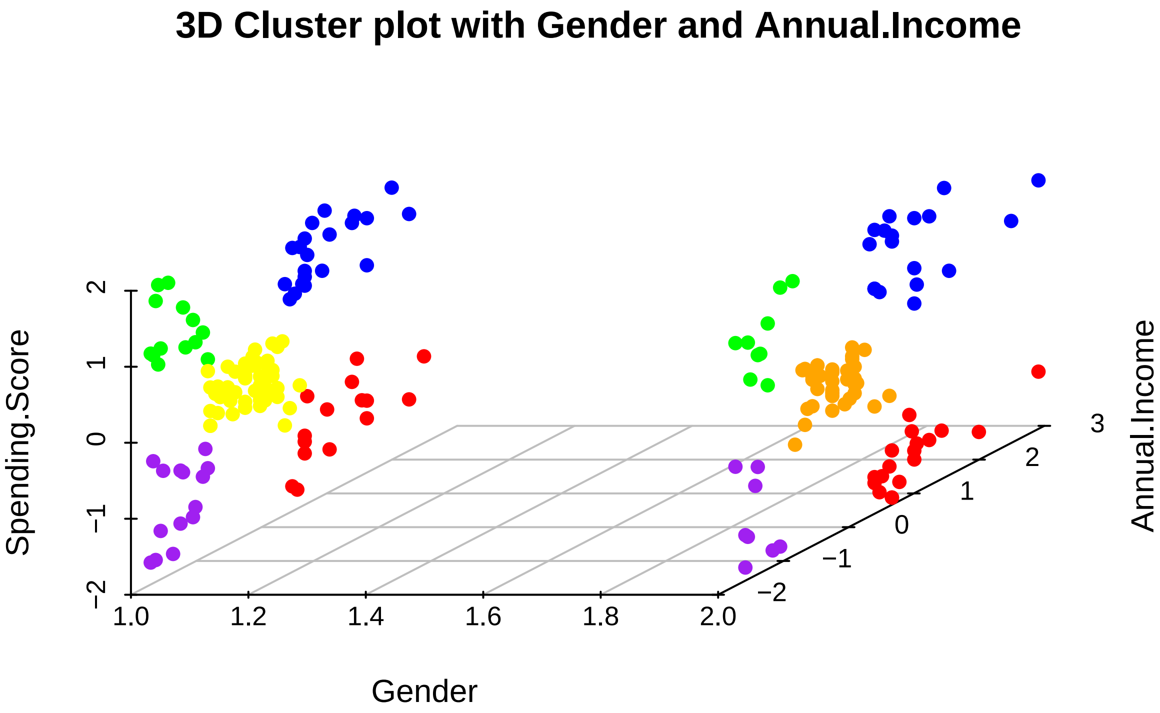
```
library(factoextra)
fviz_cluster(kmeansModel_income_score_age,
             data = Data_income_score_age,
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```



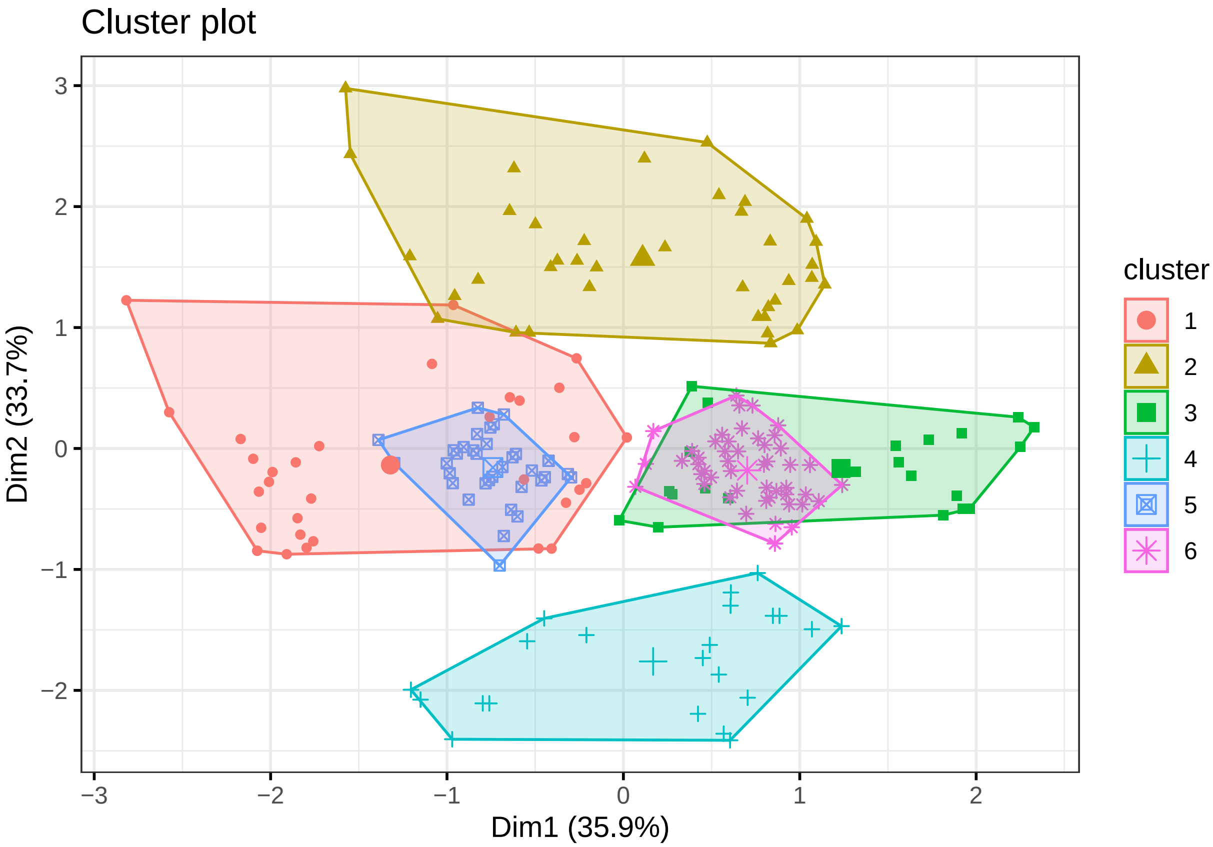
```
cat(sprintf("k = 6, the Between SS / Total SS is %.3f", kmeansModel_income_score_age$betweenss/kmeansMod
```

```
## k = 6, the Between SS / Total SS is 0.777
```

```
# Reduce age
Data_income_score_gender = df.customers[,-2]
# Data_income_score_gender = Mall_Customers1[,-2]
kmeansModel_income_score_gender= kmeans(Data_income_score_gender,6, nstart = 25)
library(scatterplot3d)
colors <- c("red", "blue", "green","purple","orange","yellow")
colors <- colors[kmeansModel_income_score_gender$cluster]
scatterplot3d(x=Data_income_score_gender,box = FALSE,color=colors,pch=16,main = "3D Cluster plot with Ge
```



```
library(factoextra)
fviz_cluster(kmeansModel_income_score_gender, data = Data_income_score_gender,
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw()
             )
```

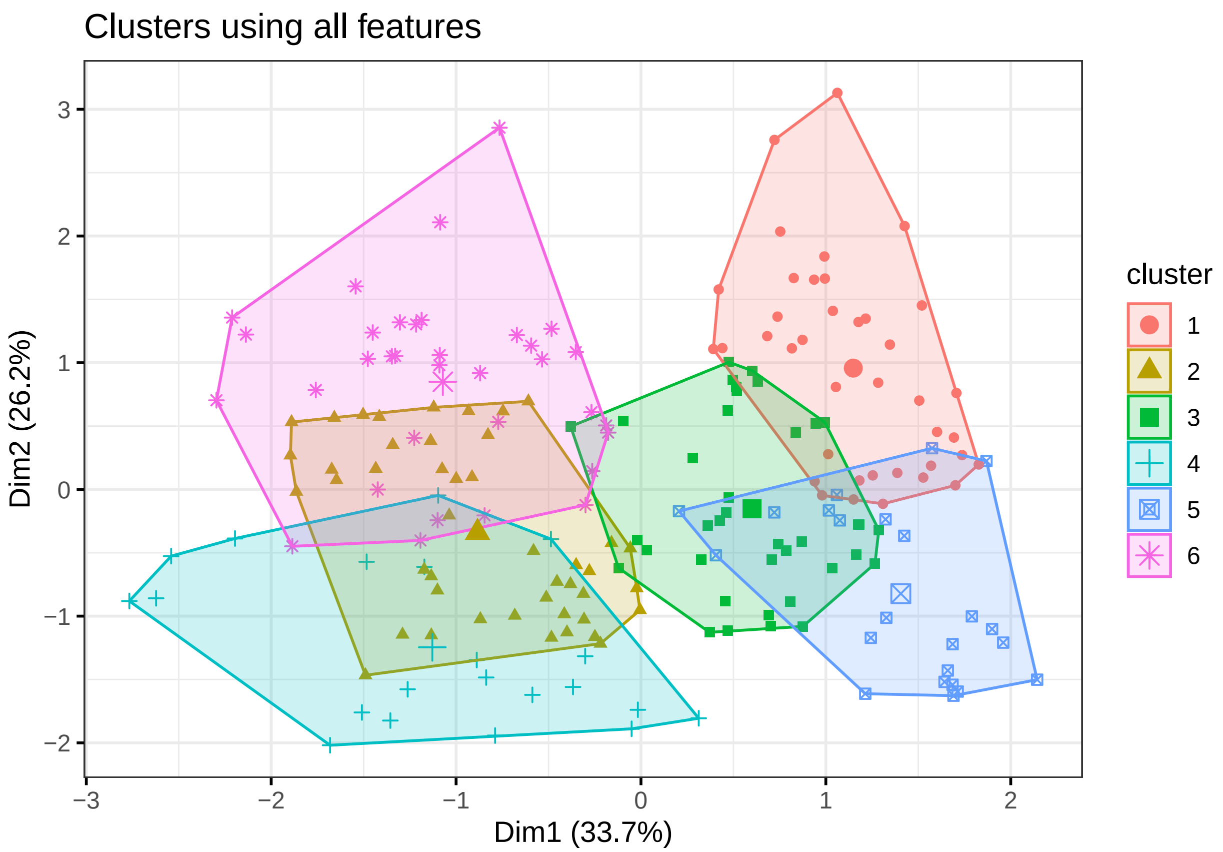


```
cat(sprintf("k = 6, the Between SS / Total SS is %.3f", kmeansModel_income_score_gender$betweenss/kmeans
```

```
## k = 6, the Between SS / Total SS is 0.790
```

Including the feature gender we split the clusters into two well differentiated groups (male and female) so we have a customer segmentation in gender but in both genders the clusters are very well differentiated so doing customer segmentation by gender could throw very good results.

```
kmeansModel_all= kmeans(df.customers, 6, nstart = 25)


library(factoextra)
fviz_cluster(kmeansModel_all, data = df.customers,
             geom = "point",
             ellipse.type = "convex",
             ggtheme = theme_bw(),
             main = "Clusters using all features"
             )
```



```
cat(sprintf("k = 6, the Between SS / Total SS is %.3f", kmeansModel_all$betweenss/kmeansModel_all$totss
```

```
## k = 6, the Between SS / Total SS is 0.719
```