# Customer Segmentation using K-Means Clustering and PCA

Tracy Whitney Akinyi

2022-07-18

In this project we are going to perform one of the applications of unsupervised machine learning - customer segmentation.In this project, we will implement customer segmentation in R.

In this project we will explore the data by performing descriptive analysis, exploratory analysis and implement different versions of K-means algorithm. Customer segmentation is the process by which you divide your customers into segments up based on common characteristics – such as demographics or behaviors, so you can market to those customers more effectively.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 data set.

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.

## *Data Exploration*

**Load libraries**

```
library(readr)
library(dlookr)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'
```

```
##
## Attaching package: 'dlookr'
```

```
## The following object is masked from 'package:base':
##
##     transform
```

```
library(cluster)
library(grid)
library(gridExtra)
library(NbClust)
library(factoextra)
```

```
## Loading required package: ggplot2
```

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

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(flextable)
library(ggstatsplot)
```

```
## You can cite this package as:
##      Patil, I. (2021). Visualizations with statistical details: The 'ggstatsplot' approach.
##      Journal of Open Source Software, 6(61), 3167, doi:10.21105/joss.03167
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v purrr   0.3.4     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::combine() masks gridExtra::combine()
## x purrr::compose() masks flextable::compose()
## x tidyr::extract() masks dlookr::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
```

***Load data set***

```
cs <- read_csv("D:/R Directories/Customer Segmentation/data/Mall_Customers.csv")
```

```
## Rows: 200 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): CustomerID, Genre
## dbl (3): Age, Annual Income (k$), Spending Score (1-100)
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
View(cs)
```

***Descriptive summaries***

```
dlookr::describe(cs,quantiles = c(.25,.50,.75)) %>% flextable()
```

```
## Warning: Warning: fonts used in ‘flextable‘ are ignored because the ‘pdflatex‘
## engine is used and not ‘xelatex‘ or ‘lualatex‘. You can avoid this warning
## by using the ‘set_flextable_defaults(fonts_ignore=TRUE)‘ command or use a
## compatible engine by defining ‘latex_engine: xelatex‘ in the YAML header of the
## R Markdown document.
```

| described_variables | n | na | mean | sd | se_mean | IQR | skewness | kurtosis |
|---|---|---|---|---|---|---|---|---|
| Age | 200 | 0 | 38.85 | 13.96901 | 0.987758 | 20.25 | 0.4855689 | -0.67157286 |
| Annual Income (k$) | 200 | 0 | 60.56 | 26.26472 | 1.857196 | 36.50 | 0.3218425 | -0.09848709 |
| Spending Score (1-100) | 200 | 0 | 50.20 | 25.82352 | 1.825999 | 38.25 | -0.0472202 | -0.82662911 |

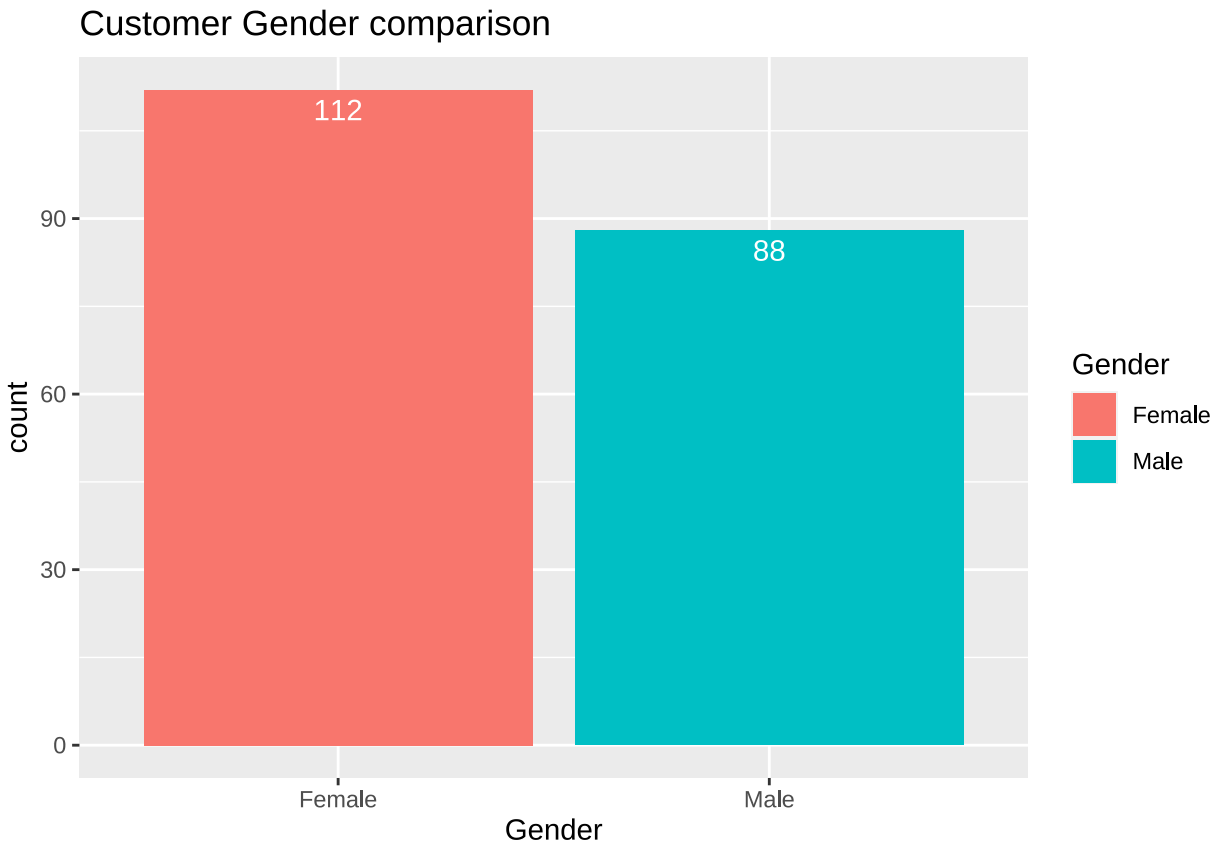
```
DF = cs[,c("Age","Annual Income (k$)","Spending Score (1-100)")]
ggpairs(DF)
```

### *Rename Genre as Gender*

```
cs <- rename(cs,Gender = Genre)
```

### *Gender distribution*

```
cs %>% ggplot(aes(x=Gender,fill = Gender))+
       geom_bar()+
     ggtitle("Customer Gender comparison")+
      geom_text(aes(label = ..count..),stat = "count", vjust = 1.5, colour = "white")
```

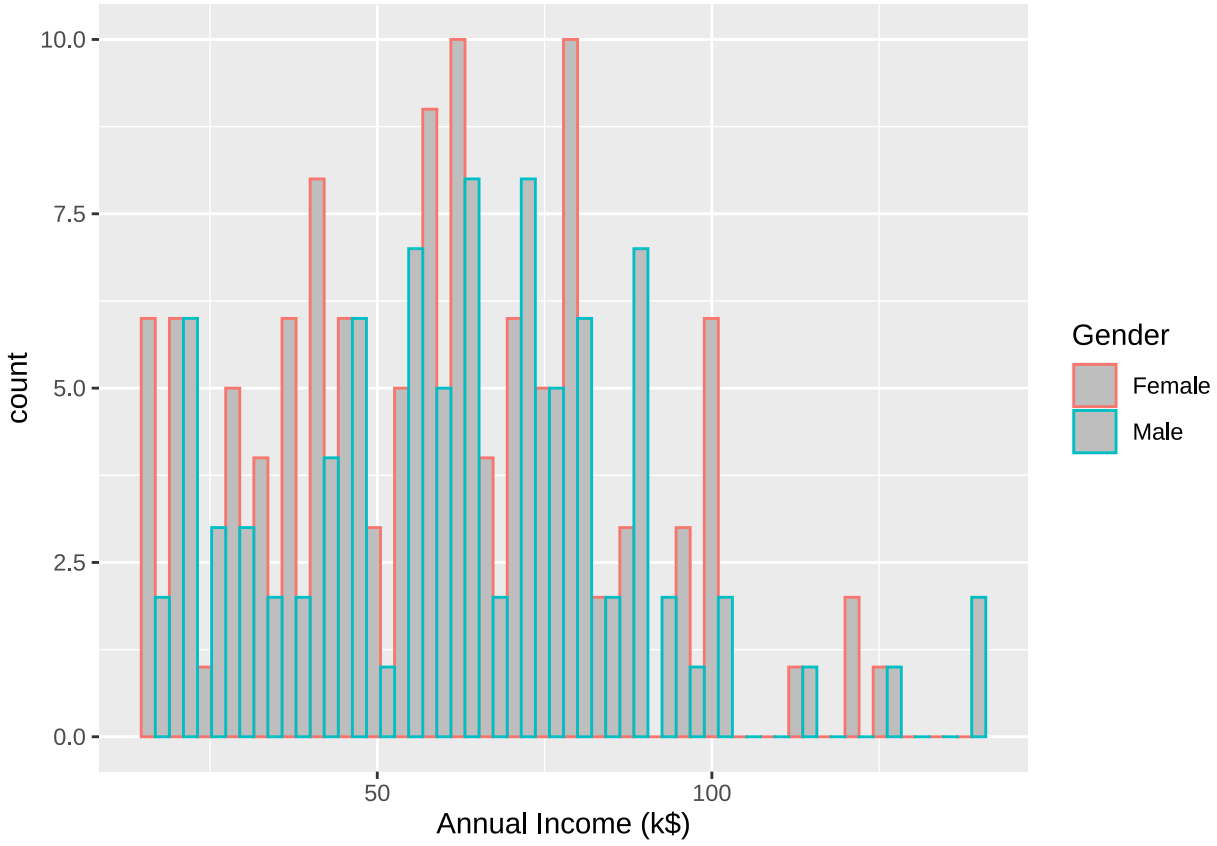



From the above bar plot we can observe that the number of females are more than males.

### *Distribution of Annual Income*

```
cs %>% ggplot(aes(x= `Annual Income (k$)`, color = Gender)) +
  geom_histogram(fill = "grey", position="dodge")
```

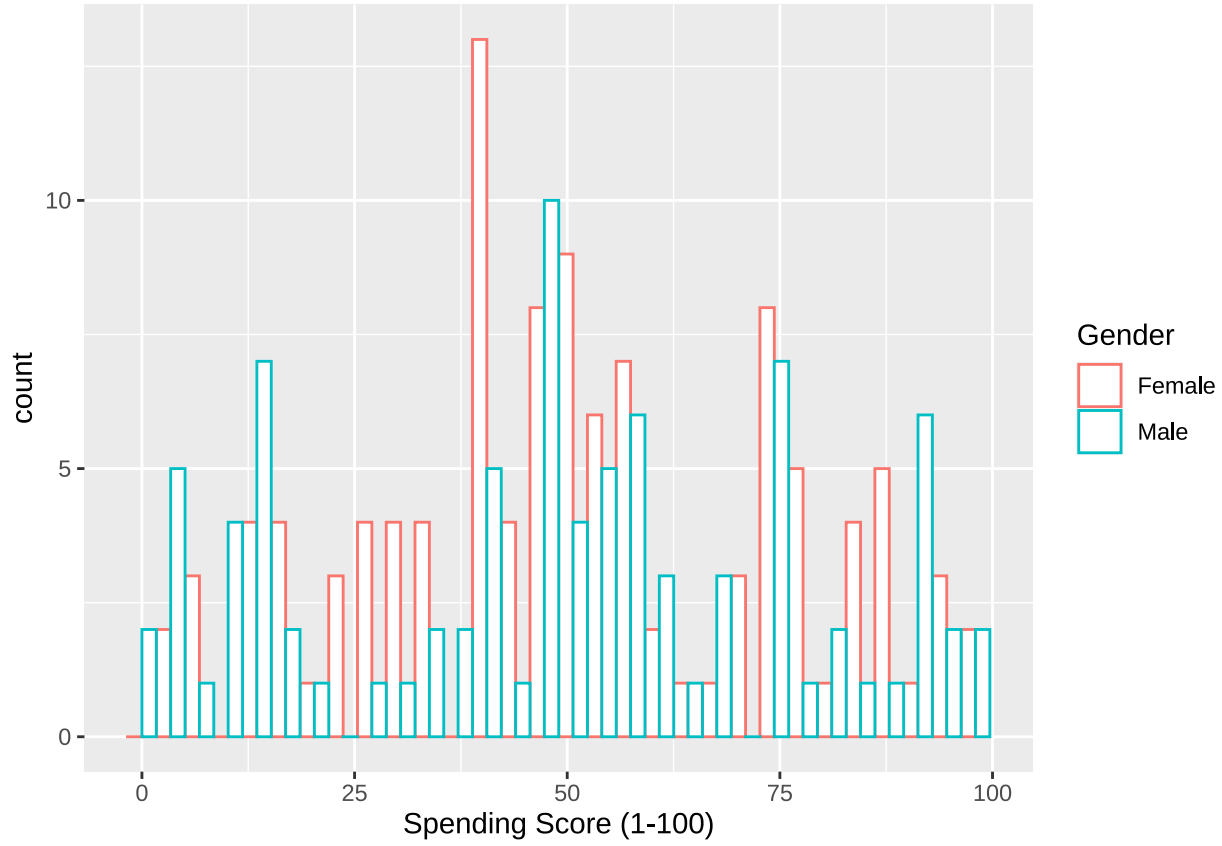
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




### *Distribution of Spending Score*

```
cs %>% ggplot(aes(x= `Spending Score (1-100)`, color = Gender)) +
  geom_histogram(fill = "white", position="dodge")
```

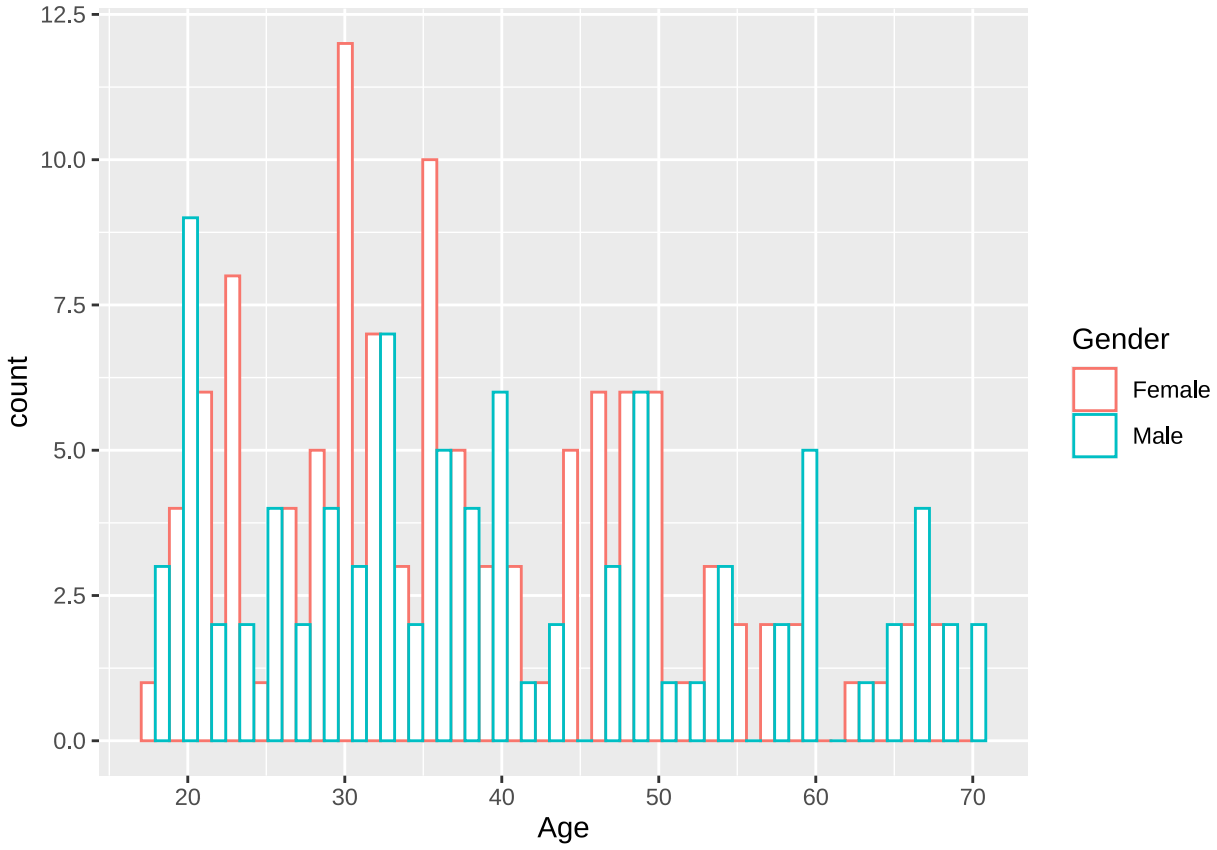
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




*## Distribution of Age*

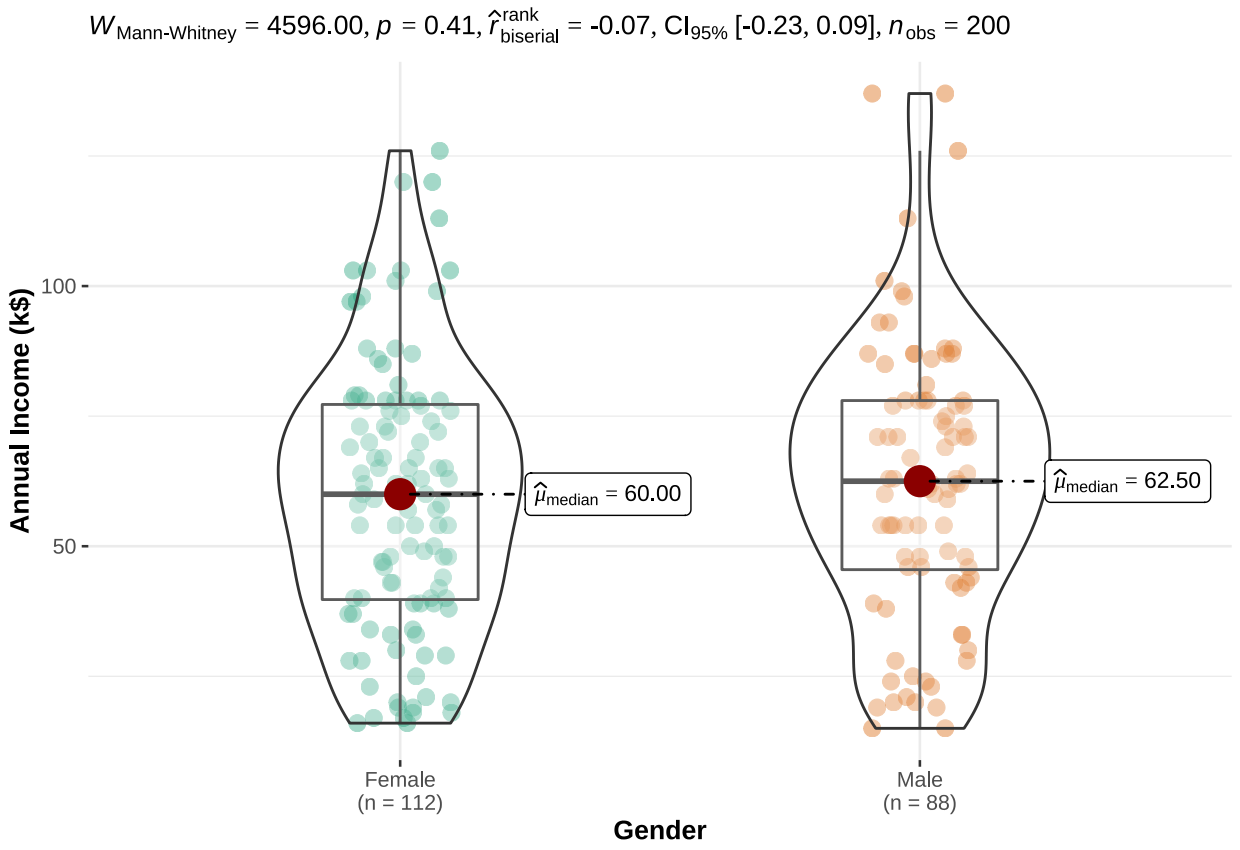
```
cs %>% ggplot(aes(x= Age, color = Gender)) +
  geom_histogram(fill = "white", position="dodge")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



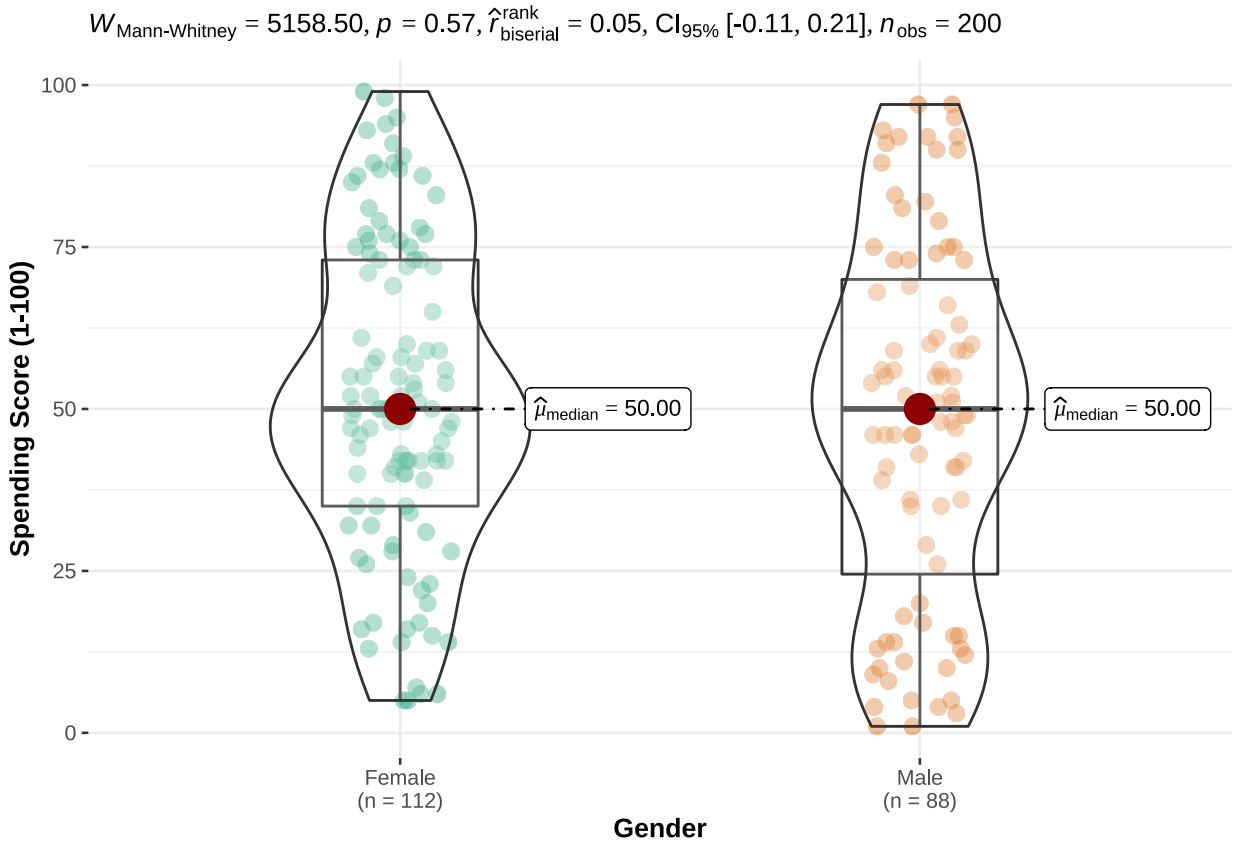

### *Comparing Gender with Annual Income*

```
cs %>% ggbetweenstats(x=Gender,y=`Annual Income (k$)`,type = "np")
```




### *Comparing Gender with Spending Score*

```
cs %>% ggbetweenstats(x=Gender,y=`Spending Score (1-100)`,type = "np")
```




The Annual Income and Spending Score between the genders doesn't differ significantly.

## *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 data set randomly that will serve as the initial centers for our clusters. This link talks more about K- means clustering in R:

https://www.datanovia.com/en/lessons/k-means-clustering-in-r-algorith-and-practical-examples/

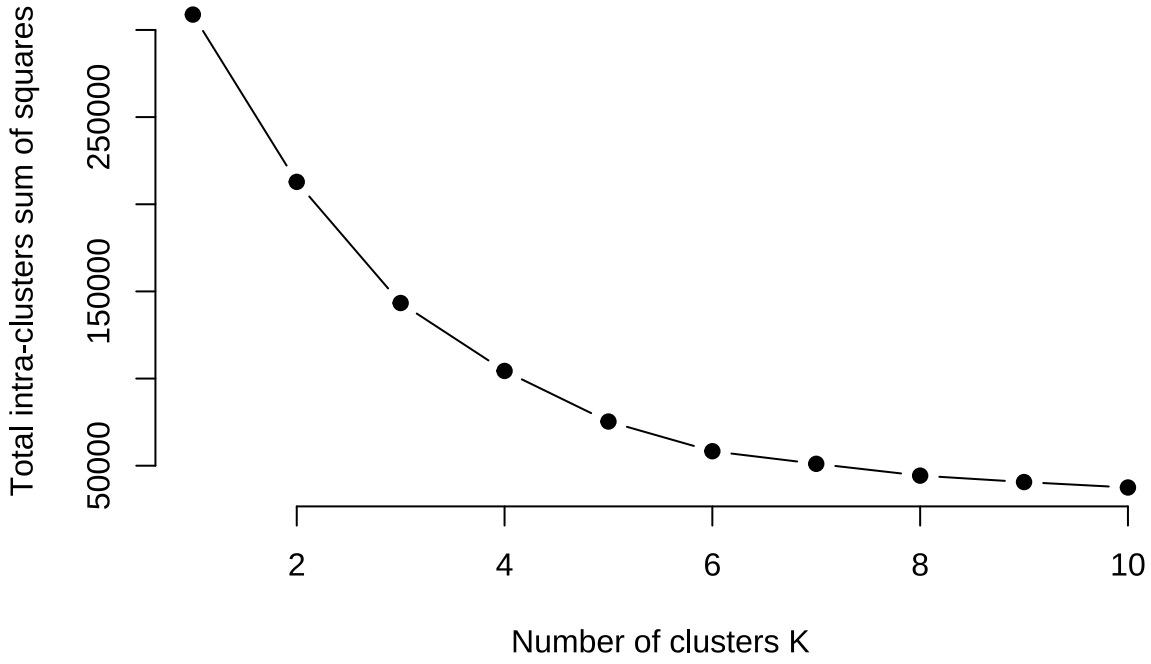
While working with clusters, you need to specify the number of clusters to use. You would like to utilize the optimal number of clusters. To help you in determining the optimal clusters, there are two popular methods –

*Elbow method
*Silhouette method

### *Elbow method*

```
set.seed(123)
# function to calculate total intra-cluster sum of square
iss <- function(k) {
  kmeans(cs[,3:5],k,iter.max=100,nstart=100,algorithm="Lloyd" )$tot.withinss
}
```

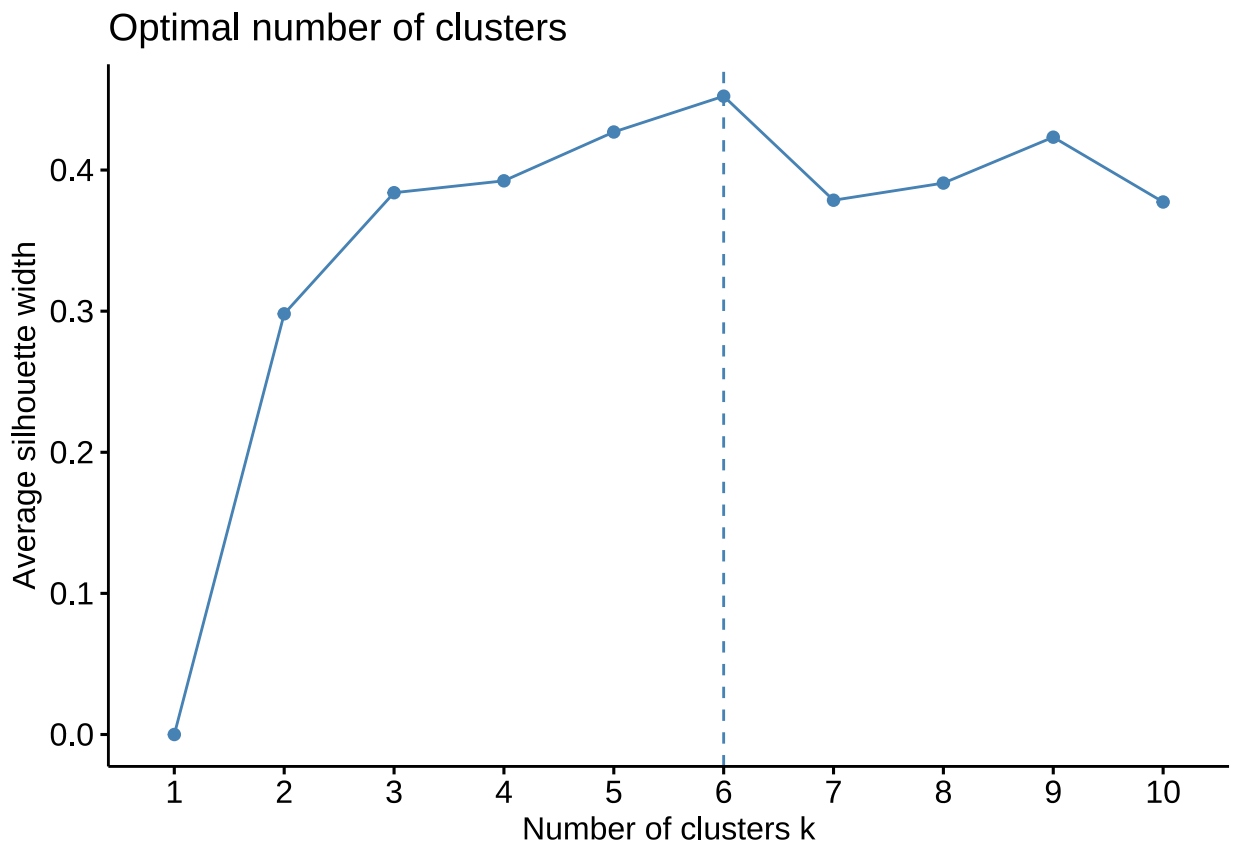
```
k.values <- 1:10


iss_values <- map_dbl(k.values, iss)

plot(k.values, iss_values,
    type="b", pch = 19, frame = FALSE,
    xlab="Number of clusters K",
    ylab="Total intra-clusters sum of squares")
```




From the above graph, we conclude that 4 appears to be the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot. Let's try the silhouette method.

***Silhouette method***

```
fviz_nbclust(cs[,3:5], kmeans, method = "silhouette")
```




From the above graph, we conclude that 6 appears is the appropriate number of clusters

**Computing gap statistic**

```
k6<-kmeans(cs[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
k6
```

```
## K-means clustering with 6 clusters of sizes 45, 21, 35, 39, 38, 22
##
## Cluster means:
##        Age Annual Income (k$) Spending Score (1-100)
## 1 56.15556           53.37778                49.08889
## 2 44.14286           25.14286                19.52381
## 3 41.68571           88.22857                17.28571
## 4 32.69231           86.53846                82.12821
## 5 27.00000           56.65789                49.13158
## 6 25.27273           25.72727                79.36364
##
## Clustering vector:
##   [1] 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2 6 2
##  [38] 6 2 6 1 6 1 5 2 6 1 5 5 5 1 5 5 1 1 1 1 1 5 1 1 5 1 1 1 5 1 1 5 5 1 1 1 1
##  [75] 1 5 1 5 5 1 1 5 1 1 5 1 1 5 5 1 1 5 1 5 5 5 1 5 1 5 5 1 1 5 1 5 1 1 1 1 1
## [112] 5 5 5 5 5 1 1 1 1 5 5 5 4 5 4 3 4 3 4 3 4 5 4 3 4 3 4 3 4 3 4 5 4 3 4 3 4
## [149] 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4 3
```

```
## [186] 4 3 4 3 4 3 4 3 4 3 4 3 4 3 4
##
## Within cluster sum of squares by cluster:
## [1]  8062.133  7732.381 16690.857 13972.359  7742.895  4099.818
##  (between_SS / total_SS =  81.1 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

***Calculating Principal Component Analysis***

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

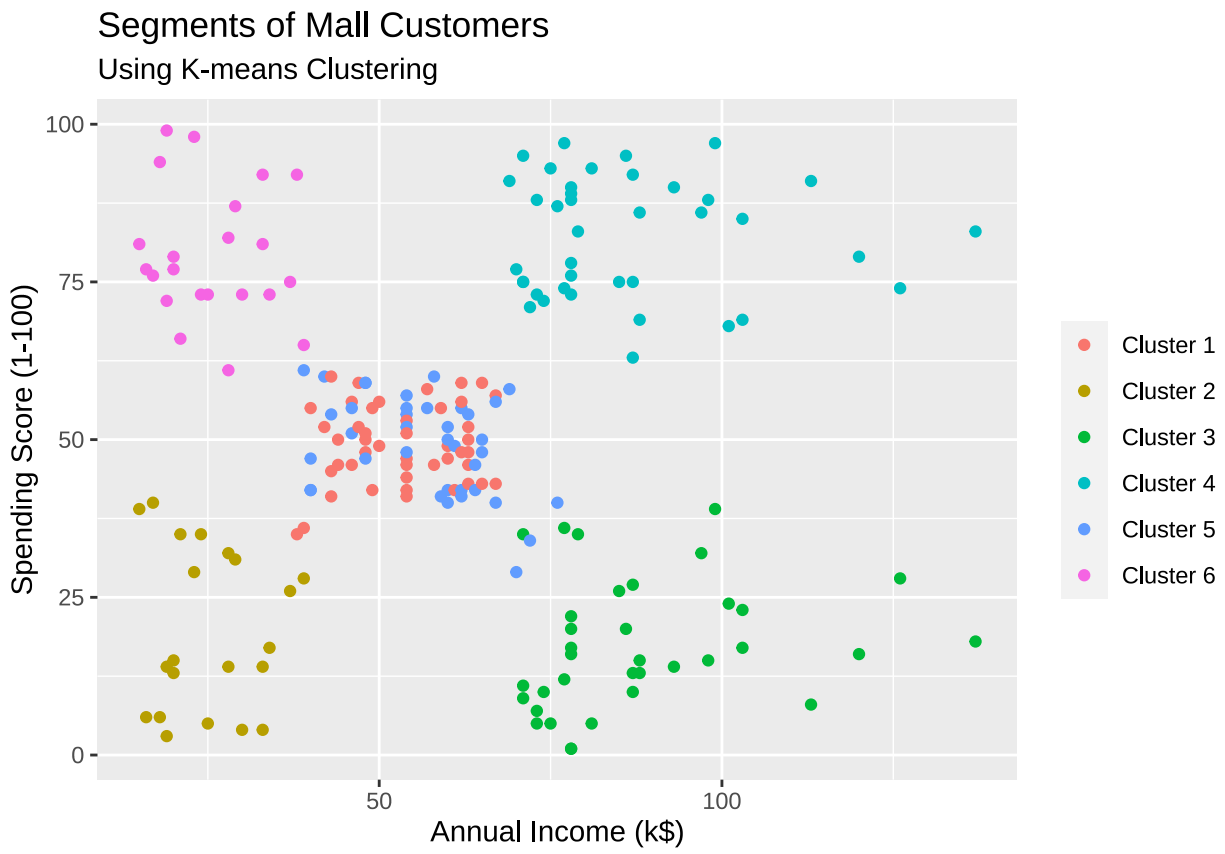
```
## Importance of components:
##                             PC1     PC2     PC3
## Standard deviation     26.4625 26.1597 12.9317
## Proportion of Variance  0.4512  0.4410  0.1078
## Cumulative Proportion   0.4512  0.8922  1.0000
```

```
pca$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
```

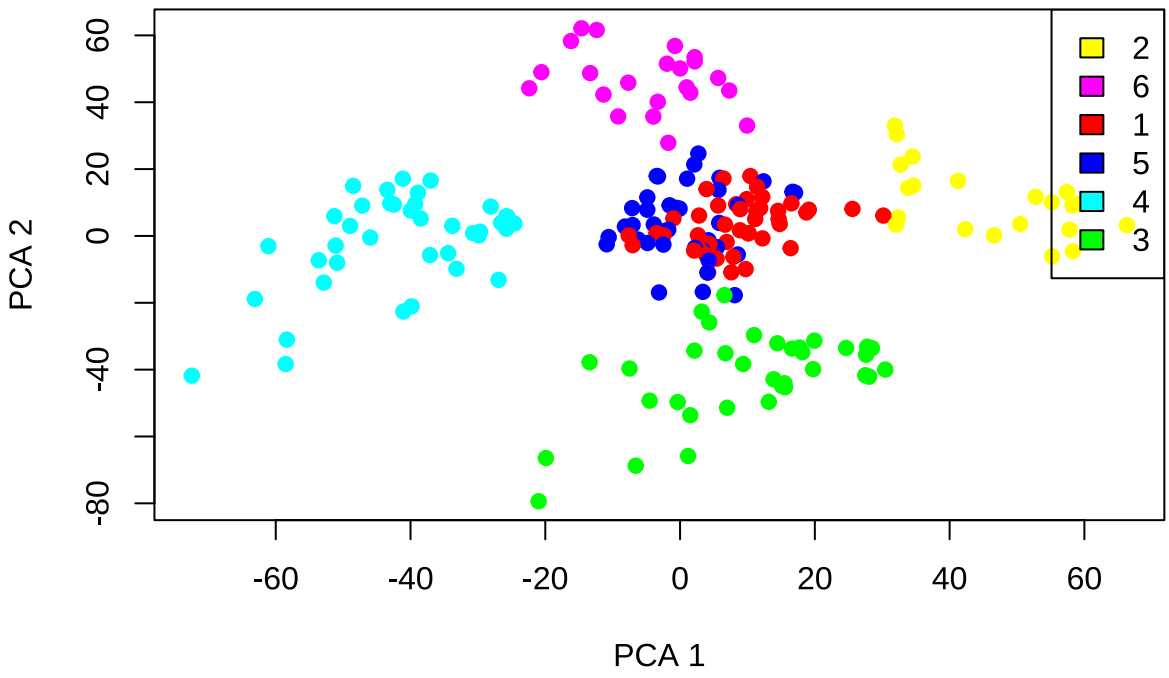
***Visualizing the clusters***

```
set.seed(1)
cs %>% ggplot(aes(x =`Annual Income (k$)`, y = `Spending Score (1-100)`)) +
      geom_point(stat = "identity", aes(color = as.factor(k6$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")
```




- Cluster 1 and 5 represent customers with medium average income and spending score.
- Cluster 2 represents customers with low annual income ans low spending score.
- Cluster 3 represents customers with high annual income and low spending score.
- Cluster 4 represents customers with high annual income and high spending score.
- Cluster 6 represents customers with low annual income and high spending score.

```
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(pca$x[,1:2], col =kCols(digCluster),pch =19,xlab ="PCA 1",ylab="PCA 2")
legend("topright",unique(dignm),fill=unique(kCols(digCluster)))
```




- Cluster 1 and 5 have medium PCA 1 and PCA 2
- Cluster 2 has a high PCA 1 and medium PCA 2
- Cluster 3 has a medium PCA 1 and low PCA 2
- Cluster 4 has a low PCA 1 and medium PCA 2
- Cluster 6 has a medium PCA 1 and high PCA 2.