## Data Science Project – Customer Segmentation using Machine Learning in R

## Discuss the business problem/goal

# Customer Segmentation is one the most important applications of unsupervised 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.

# 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

## Identify where the dataset was retrieved from

# In this machine learning project, the data was obtained from DataFlair.

## Perform data exploration

#Read input data to gain necessary insights about it.

customer\_data <- read.csv("Mall\_Customers.csv")  
str(customer\_data)

## 'data.frame': 200 obs. of 5 variables:  
## $ CustomerID : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Genre : chr "Male" "Male" "Female" "Female" ...  
## $ Age : int 19 21 20 23 31 22 35 23 64 30 ...  
## $ Annual.Income..k.. : int 15 15 16 16 17 17 18 18 19 19 ...  
## $ Spending.Score..1.100.: int 39 81 6 77 40 76 6 94 3 72 ...

names(customer\_data)

## [1] "CustomerID" "Genre" "Age"   
## [4] "Annual.Income..k.." "Spending.Score..1.100."

# Below code 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)

## CustomerID Genre Age Annual.Income..k.. Spending.Score..1.100.  
## 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

summary(customer\_data$Age)

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

# The below code displays standard deviation and descriptive stats of customer age and annual income

sd(customer\_data$Age)

## [1] 13.96901

summary(customer\_data$Annual.Income..k..)

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

sd(customer\_data$Annual.Income..k..)

## [1] 26.26472

summary(customer\_data$Age)

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

## The below code displays SD of the customer spending score

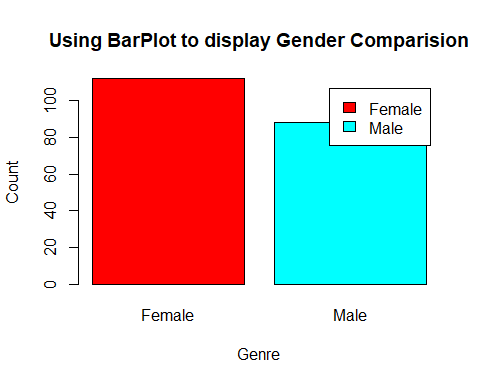
sd(customer\_data$Spending.Score..1.100.)

## [1] 25.82352

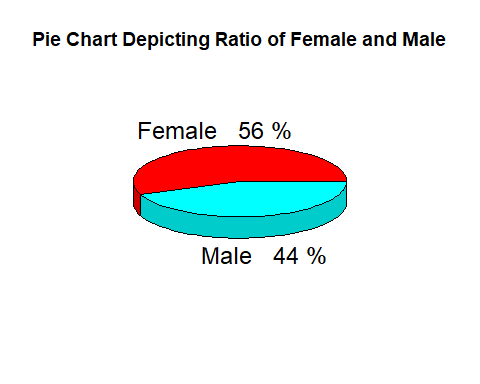
## Customer Gender Visualization

# barplot and a piechart to show the gender distribution across our customer\_data dataset

a=table(customer\_data$Genre)  
barplot(a,main="Using BarPlot to display Gender Comparision",  
 ylab="Count",  
 xlab="Genre",  
 col=rainbow(2),  
 legend=rownames(a))

 # From the above barplot, we observe that the number of females is higher than the males. Now, let us visualize a pie chart to observe the ratio of male and female distribution.

pct=round(a/sum(a)\*100)  
lbs=paste(c("Female","Male")," ",pct,"%",sep=" ")  
library(plotrix)  
pie3D(a,labels=lbs,  
 main="Pie Chart Depicting Ratio of Female and Male")

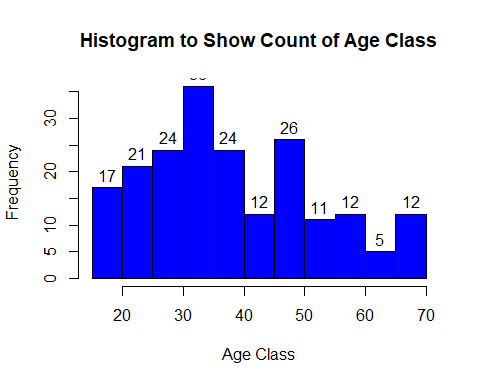
 # From the above graph, we conclude that the percentage of females is 56%, whereas the percentage of male in the customer dataset is 44%.

## Visualization of Age Distribution

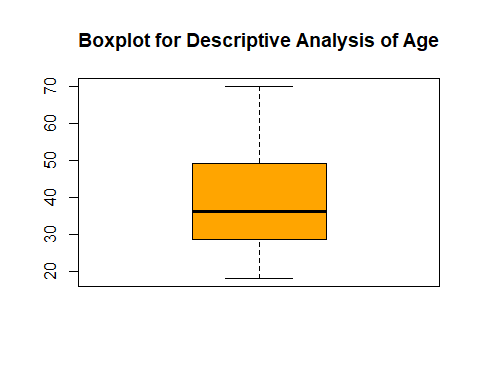
summary(customer\_data$Age)

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

hist(customer\_data$Age,  
 col="blue",  
 main="Histogram to Show Count of Age Class",  
 xlab="Age Class",  
 ylab="Frequency",  
 labels=TRUE)



boxplot(customer\_data$Age,  
 col="orange",  
 main="Boxplot for Descriptive Analysis of Age")

 # 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.

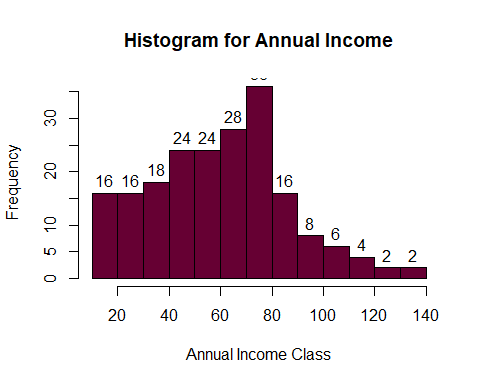
## Analysis of the Annual Income of the Customers

# 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

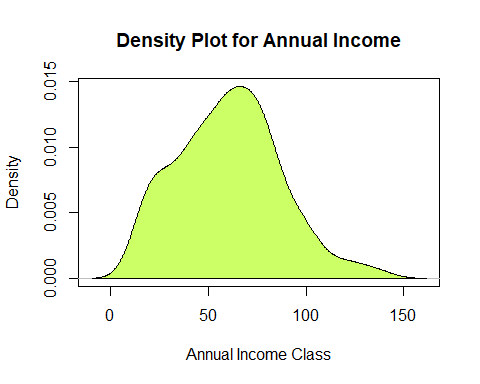
summary(customer\_data$Annual.Income..k..)

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

hist(customer\_data$Annual.Income..k..,  
 col="#660033",  
 main="Histogram for Annual Income",  
 xlab="Annual Income Class",  
 ylab="Frequency",  
 labels=TRUE)

 ## Density plot for annual income

plot(density(customer\_data$Annual.Income..k..),  
 col="yellow",  
 main="Density Plot for Annual Income",  
 xlab="Annual Income Class",  
 ylab="Density")  
polygon(density(customer\_data$Annual.Income..k..),  
 col="#ccff66")

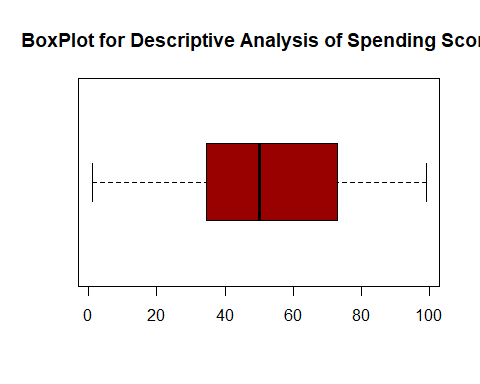
 # From the above descriptive analysis, we conclude that the minimum annual income of the customers is 15 and the maximum income is 137. People earning an average income of 70 have the highest frequency count in our histogram distribution. The average salary of all the customers is 60.56. In the Kernel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

## Analyzing Spending Score of the Customers

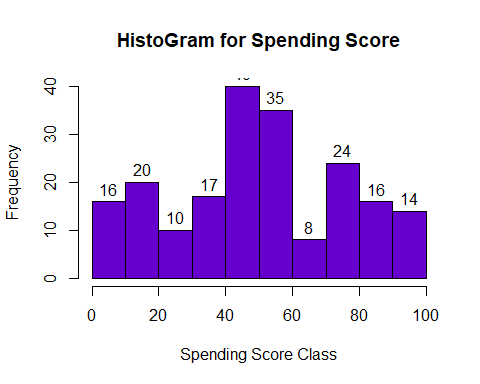
summary(customer\_data$Spending.Score..1.100.)

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

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.00 34.75 50.00 50.20 73.00 99.00  
  
boxplot(customer\_data$Spending.Score..1.100.,  
 horizontal=TRUE,  
 col="#990000",  
 main="BoxPlot for Descriptive Analysis of Spending Score")



hist(customer\_data$Spending.Score..1.100.,  
 main="HistoGram for Spending Score",  
 xlab="Spending Score Class",  
 ylab="Frequency",  
 col="#6600cc",  
 labels=TRUE)



# 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 –

# Elbow method

# Silhouette method

# Gap statistic

# Elbow Method

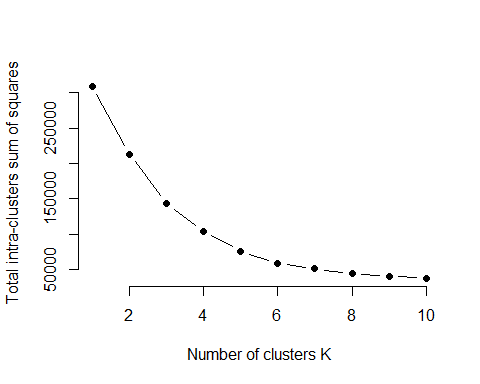
# The main goal behind cluster partitioning methods like k-means is to define the clusters such that the intra-cluster variation stays minimum.

# minimize(sum W(Ck)), k=1…k

# Where Ck represents the kth cluster and W(Ck) denotes the intra-cluster variation. With the measurement of the total intra-cluster variation, one can evaluate the compactness of the clustering boundary. We can then proceed to define the optimal clusters as follows –

# First, we calculate the clustering algorithm for several values of k. This can be done by creating a variation within k from 1 to 10 clusters. We then calculate the total intra-cluster sum of square (iss). Then, we proceed to plot iss based on the number of k clusters. This plot 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. Let us implement this in R as follows –

library(purrr)  
set.seed(123)  
# function to calculate total intra-cluster sum of square   
iss <- function(k) {  
 kmeans(customer\_data[,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")

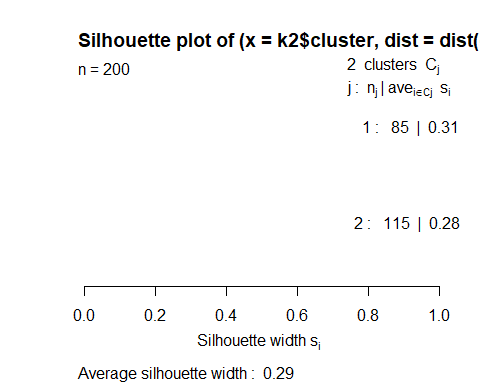
 # From the above graph, we conclude that 4 is the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot.

## Average Silhouette Method

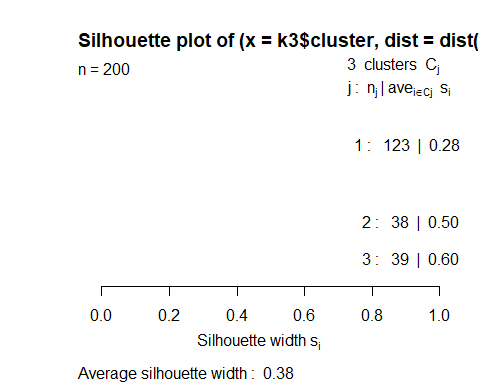
# With the help of the average silhouette method, we can measure the quality of our clustering operation. With this, we can determine how well within the cluster is the data object. If we obtain a high average silhouette width, it means that we have good clustering. The average silhouette method calculates the mean of silhouette observations for different k values. With the optimal number of k clusters, one can maximize the average silhouette over significant values for k clusters.

# Using the silhouette function in the cluster package, we can compute the average silhouette width using the kmean function. Here, the optimal cluster will possess highest average.

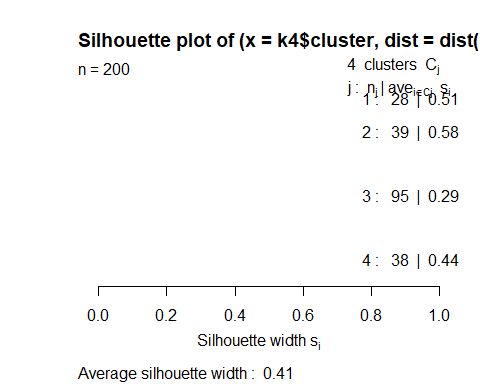
library(cluster)   
library(gridExtra)  
library(grid)  
  
  
k2<-kmeans(customer\_data[,3:5],2,iter.max=100,nstart=50,algorithm="Lloyd")  
s2<-plot(silhouette(k2$cluster,dist(customer\_data[,3:5],"euclidean")))



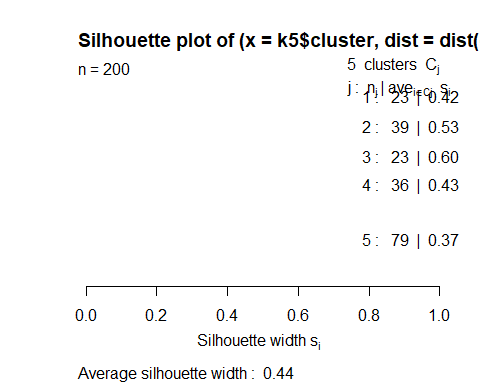
k3<-kmeans(customer\_data[,3:5],3,iter.max=100,nstart=50,algorithm="Lloyd")  
s3<-plot(silhouette(k3$cluster,dist(customer\_data[,3:5],"euclidean")))



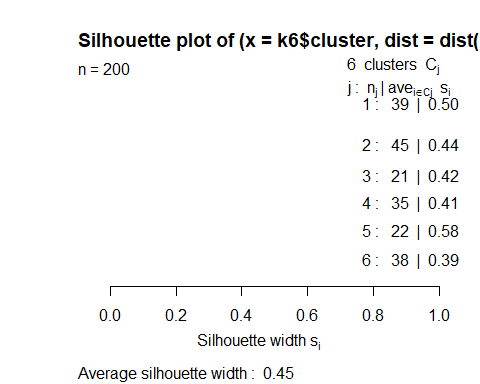
k4<-kmeans(customer\_data[,3:5],4,iter.max=100,nstart=50,algorithm="Lloyd")  
s4<-plot(silhouette(k4$cluster,dist(customer\_data[,3:5],"euclidean")))



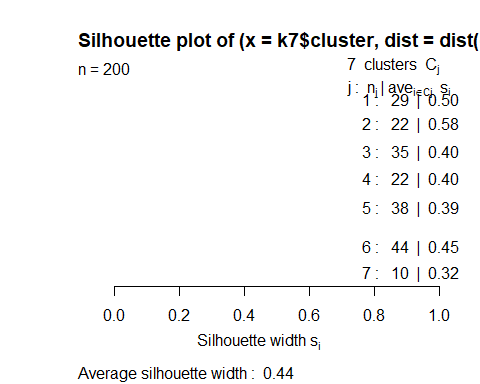
k5<-kmeans(customer\_data[,3:5],5,iter.max=100,nstart=50,algorithm="Lloyd")  
s5<-plot(silhouette(k5$cluster,dist(customer\_data[,3:5],"euclidean")))



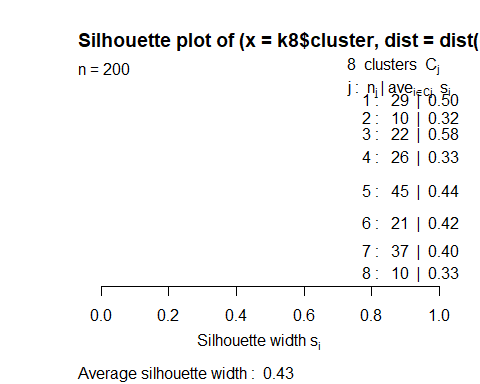
k6<-kmeans(customer\_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")  
s6<-plot(silhouette(k6$cluster,dist(customer\_data[,3:5],"euclidean")))



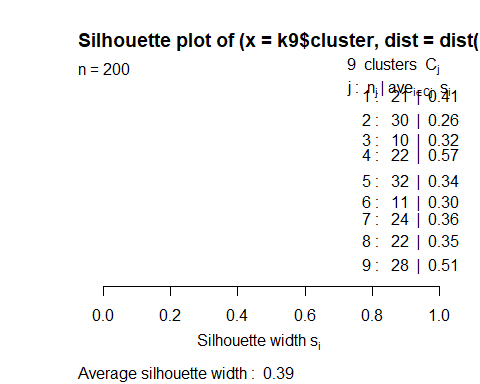
k7<-kmeans(customer\_data[,3:5],7,iter.max=100,nstart=50,algorithm="Lloyd")  
s7<-plot(silhouette(k7$cluster,dist(customer\_data[,3:5],"euclidean")))



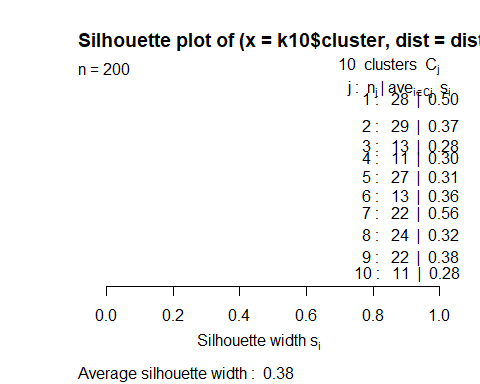
k8<-kmeans(customer\_data[,3:5],8,iter.max=100,nstart=50,algorithm="Lloyd")  
s8<-plot(silhouette(k8$cluster,dist(customer\_data[,3:5],"euclidean")))



k9<-kmeans(customer\_data[,3:5],9,iter.max=100,nstart=50,algorithm="Lloyd")  
s9<-plot(silhouette(k9$cluster,dist(customer\_data[,3:5],"euclidean")))



k10<-kmeans(customer\_data[,3:5],10,iter.max=100,nstart=50,algorithm="Lloyd")  
s10<-plot(silhouette(k10$cluster,dist(customer\_data[,3:5],"euclidean")))

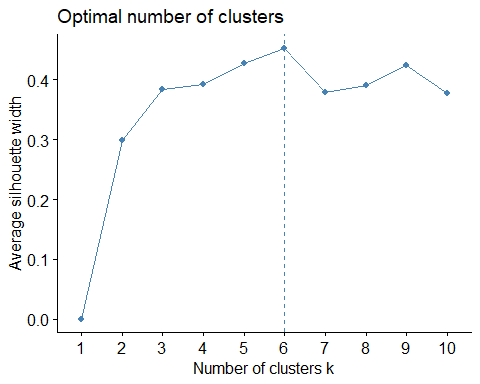
 # Now, we make use of the fviz\_nbclust() function to determine and visualize the optimal number of clusters as follows –

library(NbClust)  
library(factoextra)

## Loading required package: ggplot2

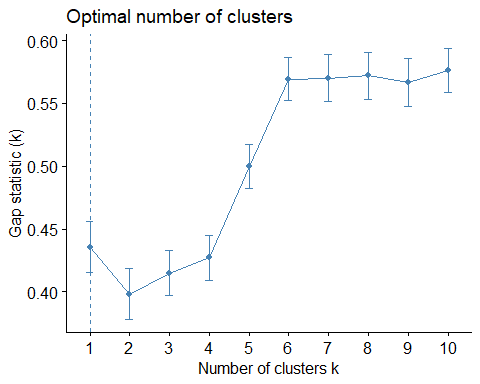
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

fviz\_nbclust(customer\_data[,3:5], kmeans, method = "silhouette")

 ## 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.

set.seed(125)  
stat\_gap <- clusGap(customer\_data[,3:5], FUN = kmeans, nstart = 25,  
 K.max = 10, B = 50)  
fviz\_gap\_stat(stat\_gap)

 # Now, let us take k = 6 as our optimal cluster –

k6<-kmeans(customer\_data[,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 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 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"

## Visualizing the Clustering Results using the First Two Principle Components

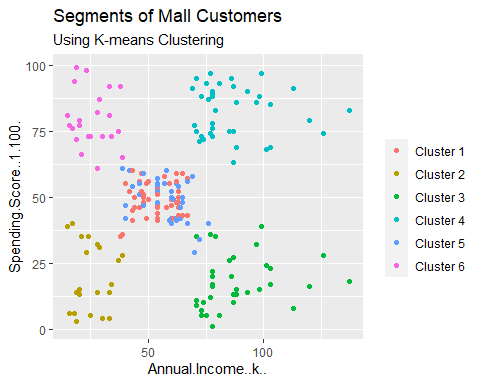
pcclust=prcomp(customer\_data[,3:5],scale=FALSE) #principal component analysis  
summary(pcclust)

## 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

set.seed(1)  
ggplot(customer\_data, 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")

 #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.

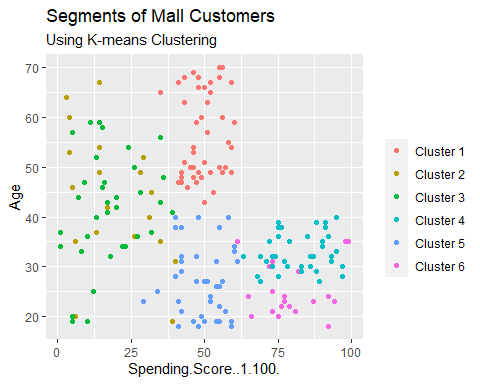
#Cluster 1 – This cluster represents the customer\_data having a high annual income as well as a high annual spend.

#Cluster 3 – This cluster denotes the customer\_data with low annual income as well as low yearly spend of income.

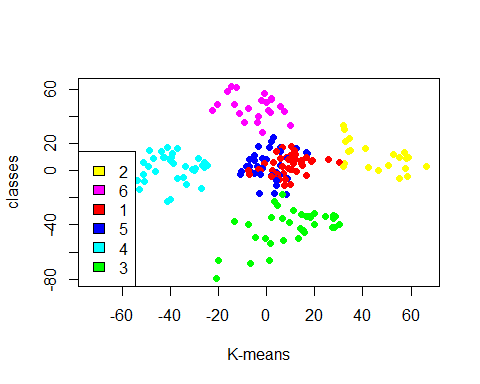
#Cluster 2 – This cluster denotes a high annual income and low yearly spend.

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

ggplot(customer\_data, aes(x =Spending.Score..1.100., y =Age)) +   
 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")



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)))



# Cluster 4 and 1 – These two clusters consist of customers with medium PCA1 and medium PCA2 score.

#Cluster 6 – This cluster represents customers having a high PCA2 and a low PCA1.

#Cluster 5 – In this cluster, there are customers with a medium PCA1 and a low PCA2 score.

#Cluster 3 – This cluster comprises of customers with a high PCA1 income and a high PCA2.

#Cluster 2 – This comprises of customers with a high PCA2 and a medium annual spend of income.

#With the help of clustering, we can understand the variables much better, prompting us to take careful decisions. With the identification of customers, companies can release products and services that target customers based on several parameters like income, age, spending patterns, etc. Furthermore, more complex patterns like product reviews are taken into consideration for better segmentation.

## 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.