RAMAIAH
Institute of Technology

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

Report on

Data Science Mini Project : Customer Segmentation

Submitted in partial fulfilment of the CIE for the

subject **Data Science and Data Science Lab(IS71 &**

ISL76) By

Gaurav Gupta - 1MS19IS040

Gaurav Sood - 1MS19IS041

Mohmed Hasan Beg - 1MS19IS070

Amir Soneji - 1MS19IS071

Under the guidance

Of

Prof. Savita K Shetty

Assistant Professor

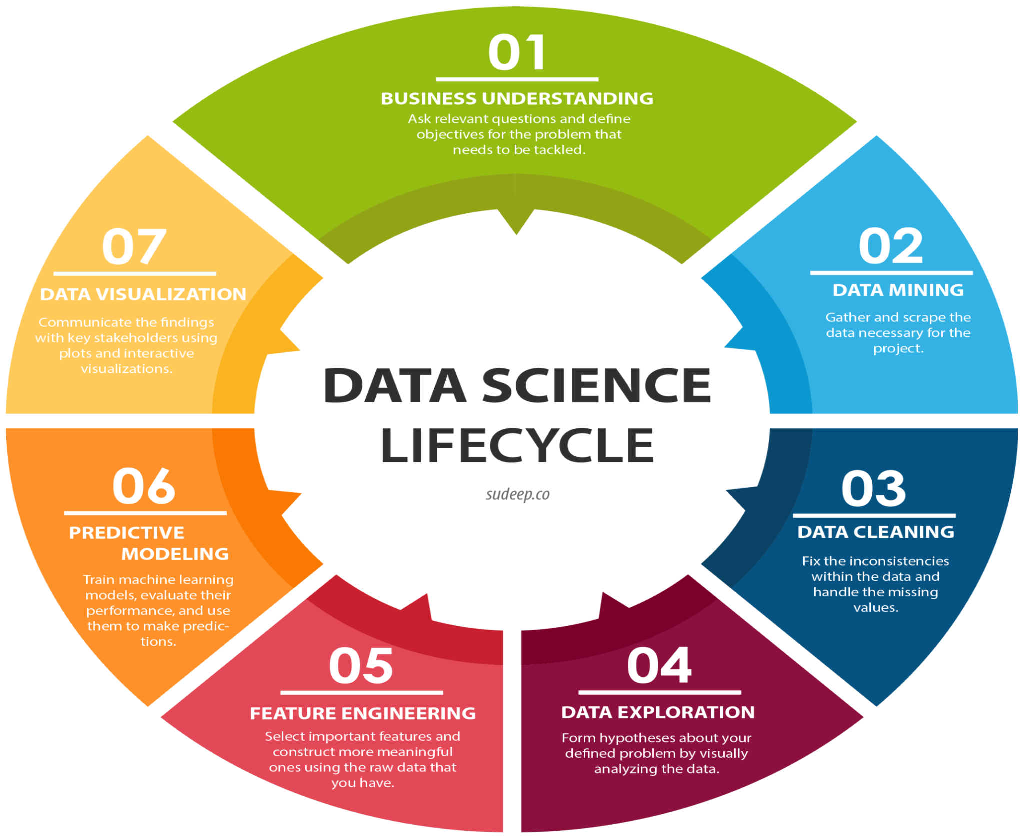
Department of ISE,

MSRIT Bengaluru –

560054

PROJECT OVERVIEW

In this Data Science R Project series, we will perform one of the most essential applications of machine learning – Customer Segmentation.
In this project, we will implement customer segmentation in R.
Whenever you need to find your best customer, customer segmentation is the ideal methodology.
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.


01
BUSINESS UNDERSTANDING
Ask relevant questions and define objectives for the problem that needs to be tackled.
02
DATA MINING
Gather and scrape the data necessary for the project.
03
DATA CLEANING
Fix the inconsistencies within the data and handle the missing values.
04
DATA EXPLORATION
Form hypotheses about your defined problem by visually analyzing the data.
05
FEATURE ENGINEERING
Select important features and construct more meaningful ones using the raw data that you have.
06
PREDICTIVE MODELING
Train machine learning models, evaluate their performance, and use them to make predictions.
07
DATA VISUALIZATION
Communicate the findings with key stakeholders using plots and interactive visualizations.
DATA SCIENCE LIFECYCLE
sudeep.co

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.

DIFFERENT DATA SET

Four datasets were provided as csv files. There were two additional files that contained information about some of the features

- Udacity_AZDIAS_052018.csv: Demographics data for the general population of Germany; 891 211 persons (rows) x 366 features (columns)
- Udacity_CUSTOMERS_052018.csv: Demographics data for customers of a mail-order company; 191 652 persons (rows) x 369 features (columns)
- Udacity_MAILOUT_052018_TRAIN.csv: Demographics data for individuals who were targets of a marketing campaign; 42 982 persons (rows) x 367 (columns)
- Udacity_MAILOUT_052018_TEST.csv: Demographics data for individuals who were targets of a marketing campaign; 42 833 persons (rows) x 366 (columns)
- DIAS Attributes — Values 2017.xlsx: Feature Information
- DIAS Information Levels — Attributes 20172017.xlsx: Feature Information
- Mall_Customers.csv - Data for individual having Annual income and Spending Score per person.

Drawbacks of using raw data :

- Raw data can often be out-of-date, denormalized, or poorly structured
- There is no built-in capacity for consistency, version control, and collaboration
- All-in-one solutions are often black boxes

IMPLEMENTING 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.

METHODOLOGY

The language used is R.

Libraries used are as follows:

library(Dplyr): Dplyr is mainly used for data manipulation in R. Dplyr is actually built around these 5 functions. These functions make up the majority of the data manipulation you tend to do. You can work with local data frames as well as with remote database tables.

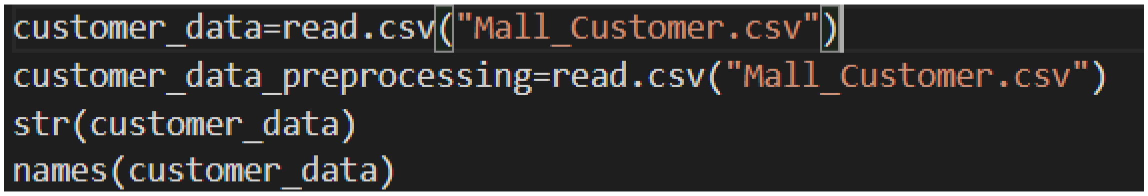
library(ggplot2): ggplot2 is a plotting package that provides helpful commands to create complex plots from data in a data frame. It provides a more programmatic interface for specifying what variables to plot, how they are displayed, and general visual properties.

library(tidylr):`tidyl` is a one such package which was built for the sole purpose of simplifying the process of creating tidy data. This tutorial provides you with the basic understanding of the four fundamental functions of data tidying that tidyr provides:

- `gather()` makes "wide" data longer
- `spread()` makes "long" data wider
- `separate()` splits a single column into multiple columns
- `unite()` combines multiple columns into a single column

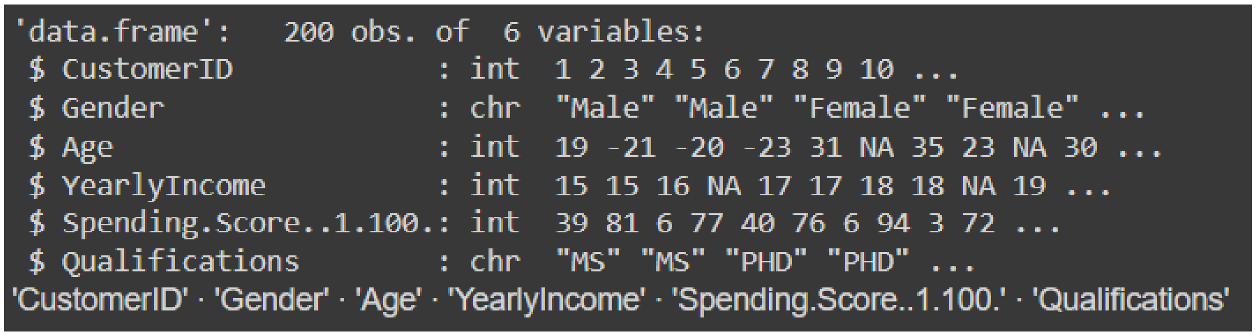
library(plotrix): Lots of plots, various labeling, axis and color scaling functions

IMPORTING DATA SET

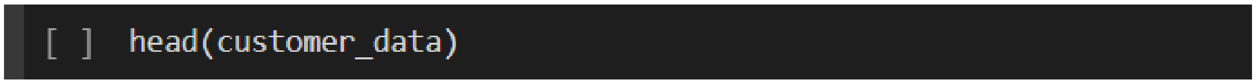


```
customer_data=read.csv("Mall_Customer.csv")
customer_data_preprocessing=read.csv("Mall_Customer.csv")
str(customer_data)
names(customer_data)
```

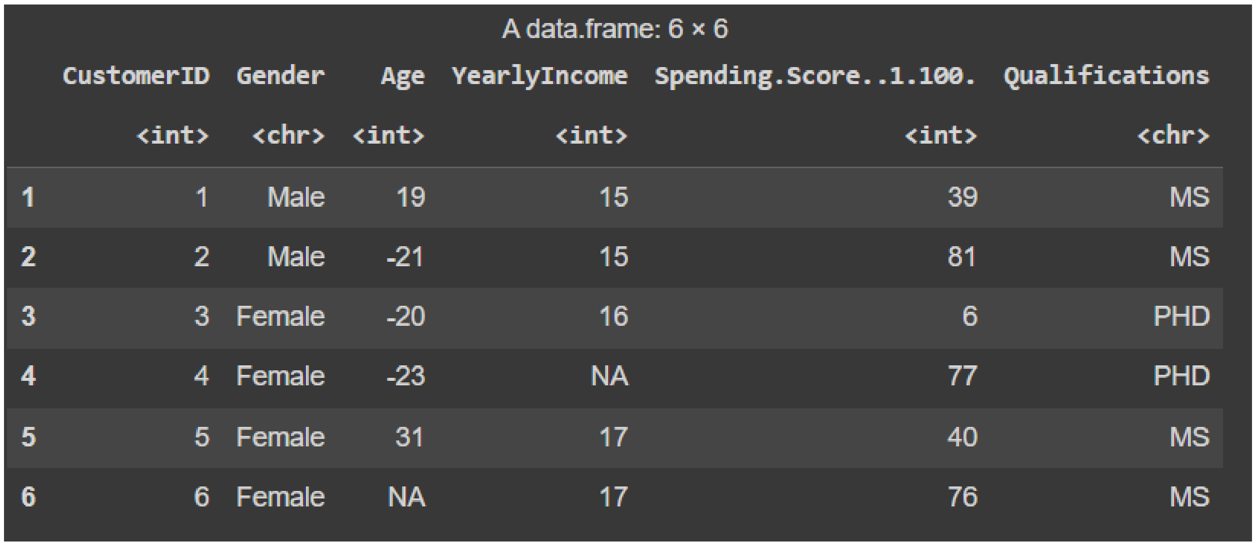
BASIC INFORMATION ABOUT DATASET



```
'data.frame':   200 obs. of  6 variables:
 $ CustomerID            : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender                : chr  "Male" "Male" "Female" "Female" ...
 $ Age                   : int  19 -21 -20 -23 31 NA 35 23 NA 30 ...
 $ YearlyIncome          : int  15 15 16 NA 17 17 18 18 NA 19 ...
 $ Spending.Score..1.100.: int  39 81 6 77 40 76 6 94 3 72 ...
 $ Qualifications        : chr  "MS" "MS" "PHD" "PHD" ...
'CustomerID' · 'Gender' · 'Age' · 'YearlyIncome' · 'Spending.Score..1.100.' · 'Qualifications'
```



```
[ ]  head(customer_data)
```



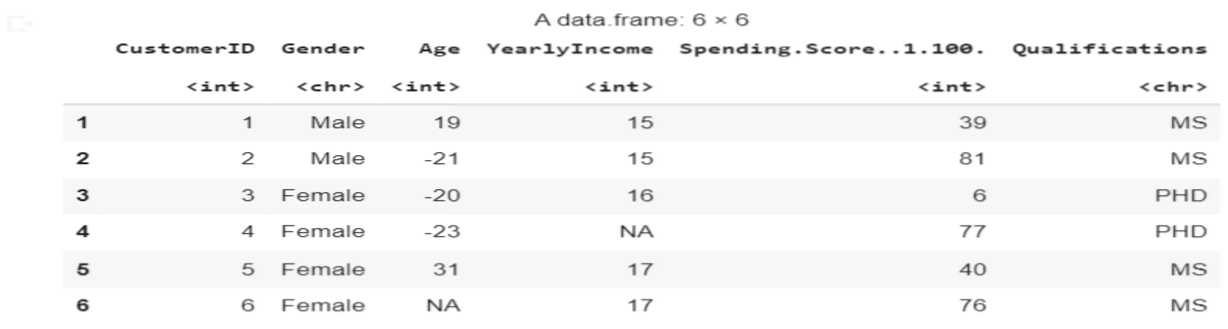
A data.frame: 6 × 6

	CustomerID	Gender	Age	YearlyIncome	Spending.Score..1.100.	Qualifications
	<int>	<chr>	<int>	<int>	<int>	<chr>
1	1	Male	19	15	39	MS
2	2	Male	-21	15	81	MS
3	3	Female	-20	16	6	PHD
4	4	Female	-23	NA	77	PHD
5	5	Female	31	17	40	MS
6	6	Female	NA	17	76	MS

DATA PREPROCESSING

NULL VALUES

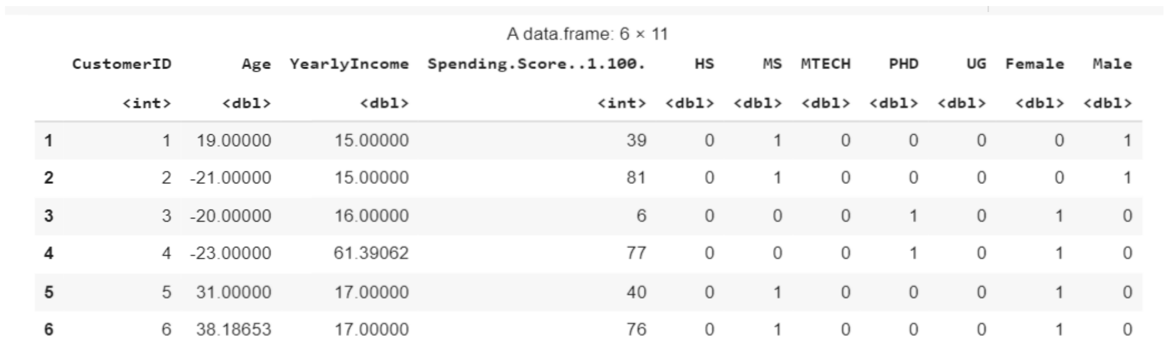
The problem of missing value is quite common in many real-life datasets. Missing value can bias the results of the machine learning models and/or reduce the accuracy of the model.
Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset.
Below is the missing data from the customer dataset.You can see the columns 'Age' and 'YearlyIncome' have some missing values.



A data.frame: 6 × 6

	CustomerID	Gender	Age	YearlyIncome	Spending.Score..1.100.	Qualifications
	<int>	<chr>	<int>	<int>	<int>	<chr>
1	1	Male	19	15	39	MS
2	2	Male	-21	15	81	MS
3	3	Female	-20	16	6	PHD
4	4	Female	-23	NA	77	PHD
5	5	Female	31	17	40	MS
6	6	Female	NA	17	76	MS

REMOVING NULL VALUES

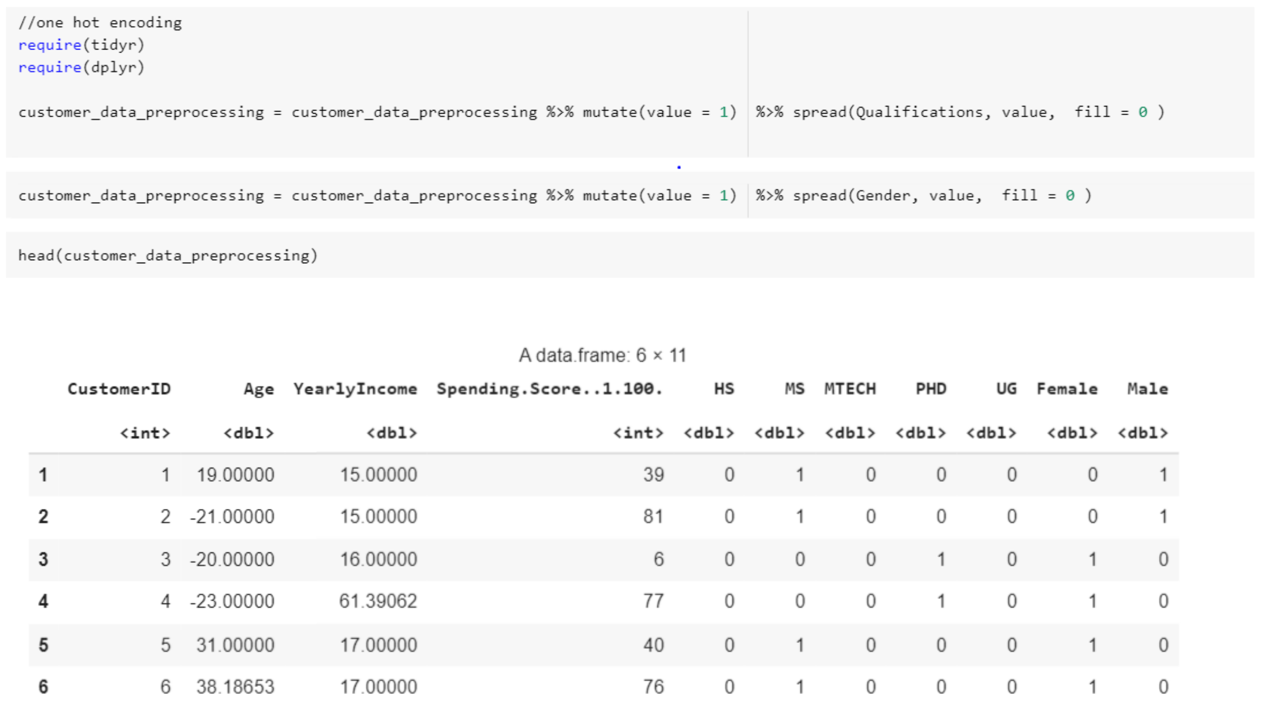


A data.frame: 6 × 11

	CustomerID	Age	YearlyIncome	Spending.Score..1.100.	HS	MS	MTECH	PHD	UG	Female	Male
	<int>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1	19.00000	15.00000	39	0	1	0	0	0	0	1
2	2	-21.00000	15.00000	81	0	1	0	0	0	0	1
3	3	-20.00000	16.00000	6	0	0	0	1	0	1	0
4	4	-23.00000	61.39062	77	0	0	0	1	0	1	0
5	5	31.00000	17.00000	40	0	1	0	0	0	1	0
6	6	38.18653	17.00000	76	0	1	0	0	0	1	0

ONE HOT ENCODING

For categorical variables where no such ordinal relationship exists, the integer encoding is not enough.In fact, using this encoding and allowing the model to assume a natural ordering between categories may result in poor performance or unexpected results (predictions halfway between categories).In this case, a one-hot encoding can be applied to the integer representation. This is where the integer encoded variable is removed and a new binary variable is added for each unique integer value.



```
//one hot encoding
require(tidyr)
require(dplyr)

customer_data_preprocessing = customer_data_preprocessing %>% mutate(value = 1) %>% spread(Qualifications, value,  fill = 0 )
```

```
customer_data_preprocessing = customer_data_preprocessing %>% mutate(value = 1) %>% spread(Gender, value,  fill = 0 )
```

```
head(customer_data_preprocessing)
```

A data.frame: 6 × 11

	CustomerID	Age	YearlyIncome	Spending.Score..1.100.	HS	MS	MTECH	PHD	UG	Female	Male
	<int>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1	19.00000	15.00000	39	0	1	0	0	0	0	1
2	2	-21.00000	15.00000	81	0	1	0	0	0	0	1
3	3	-20.00000	16.00000	6	0	0	0	1	0	1	0
4	4	-23.00000	61.39062	77	0	0	0	1	0	1	0
5	5	31.00000	17.00000	40	0	1	0	0	0	1	0
6	6	38.18653	17.00000	76	0	1	0	0	0	1	0

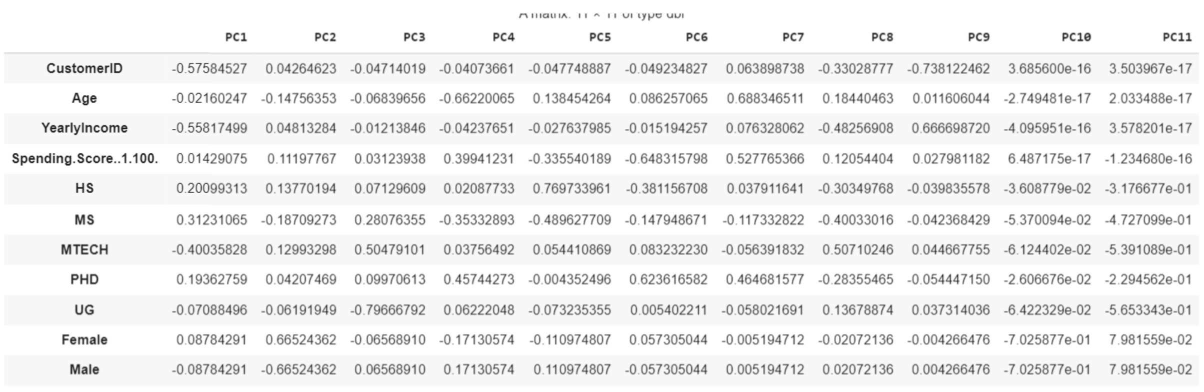
PRINCIPAL COMPONENT ANALYSIS

Principal component analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process.

So, to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.

```
#calculate principal components
results <- prcomp(customer_data_preprocessing, scale = TRUE)

#reverse the signs
results$rotation <- -1*results$rotation

#display principal components
results$rotation
```

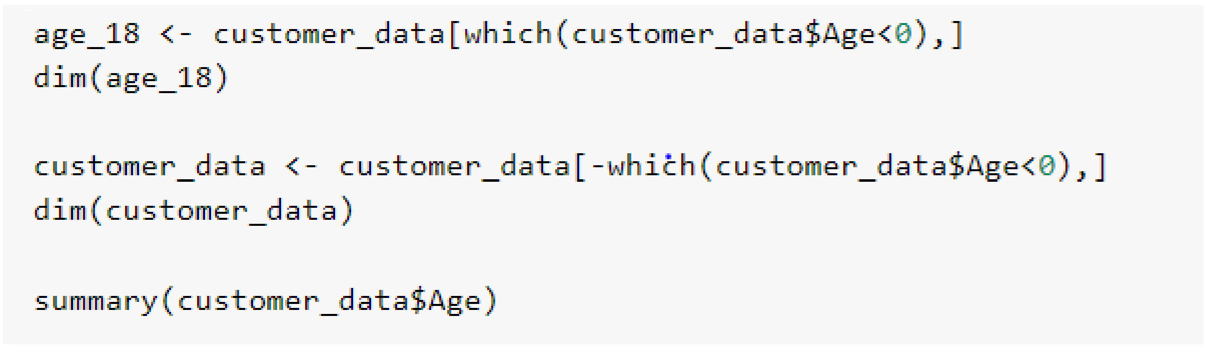


A matrix: 11 × 11 of type dbl

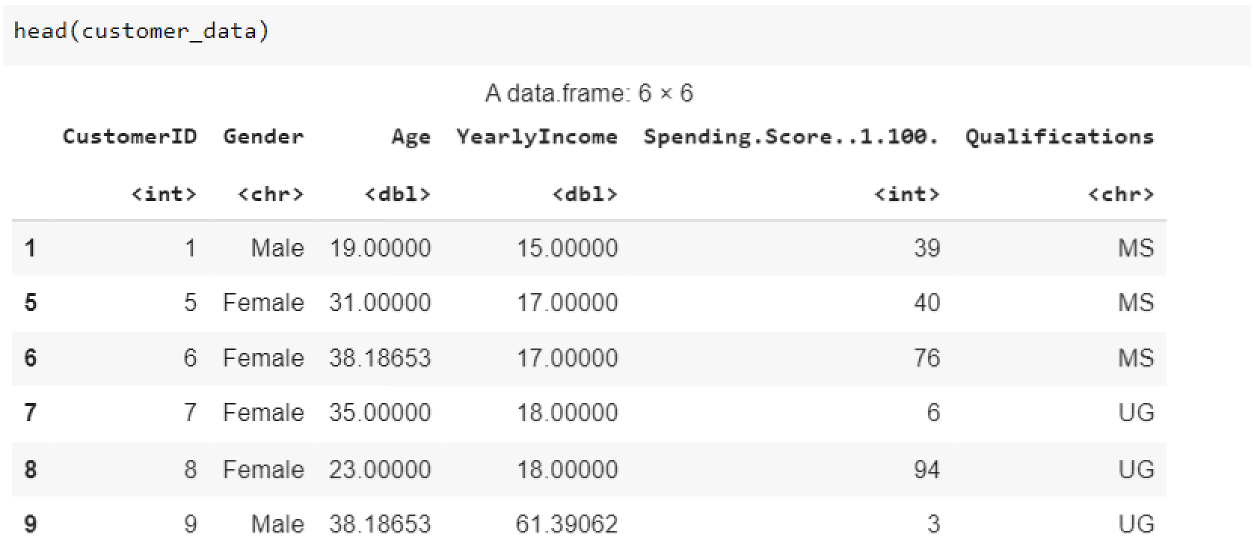
	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11
CustomerID	-0.57584527	0.04264623	-0.04714019	-0.04073661	-0.047748887	-0.049234827	0.063898738	-0.33028777	-0.738122462	3.685600e-16	3.503967e-17
Age	-0.02160247	-0.14756353	-0.06839656	-0.66220065	0.138454264	0.086257065	0.688346511	0.18440463	0.011606044	-2.749481e-17	2.033488e-17
YearlyIncome	-0.55817499	0.04813284	-0.01213846	-0.04237651	-0.027637985	-0.015194257	0.076328062	-0.48256908	0.666698720	-4.095951e-16	3.578201e-17
Spending.Score..1.100.	0.01429075	0.11197767	0.03123938	0.39941231	-0.335540189	-0.648315798	0.527765366	0.12054404	0.027981182	6.487175e-17	-1.234680e-16
HS	0.20099313	0.13770194	0.07129609	0.02087733	0.769733961	-0.381156708	0.037911641	-0.30349768	-0.039835578	-3.608779e-02	-3.176677e-01
MS	0.31231065	-0.18709273	0.28076355	-0.35332893	-0.489627709	-0.147948671	-0.117332822	-0.40033016	-0.042368429	-5.370094e-02	-4.727099e-01
MTECH	-0.40035828	0.12993298	0.50479101	0.03756492	0.054410869	0.083232230	-0.056391832	0.50710246	0.044667755	-6.124402e-02	-5.391089e-01
PHD	0.19362759	0.04207469	0.09970613	0.45744273	-0.004352496	0.623616582	0.464681577	-0.28355465	-0.054447150	-2.606676e-02	-2.294562e-01
UG	-0.07088496	-0.06191949	-0.79666792	0.06222048	-0.073235355	0.005402211	-0.058021691	0.13678874	0.037314036	-6.422329e-02	-5.653343e-01
Female	0.08784291	0.66524362	-0.06568910	-0.17130574	-0.110974807	0.057305044	-0.005194712	-0.02072136	-0.004266476	-7.025877e-01	7.981559e-02
Male	-0.08784291	-0.66524362	0.06568910	0.17130574	0.110974807	-0.057305044	0.005194712	0.02072136	0.004266476	-7.025877e-01	7.981559e-02

REMOVING NEGATIVE VALUES

Removing negative values from our dataset,there are negative values in age attribute that are to be removed by the condition of age being bigger than 0.



```r
age_18 <- customer_data[which(customer_data$Age<0),]
dim(age_18)

customer_data <- customer_data[-which(customer_data$Age<0),]
dim(customer_data)

summary(customer_data$Age)
```



```r
head(customer_data)
```

A data.frame: 6 × 6

	CustomerID	Gender	Age	YearlyIncome	Spending.Score..1.100.	Qualifications
	<int>	<chr>	<dbl>	<dbl>	<int>	<chr>
1	1	Male	19.00000	15.00000	39	MS
5	5	Female	31.00000	17.00000	40	MS
6	6	Female	38.18653	17.00000	76	MS
7	7	Female	35.00000	18.00000	6	UG
8	8	Female	23.00000	18.00000	94	UG
9	9	Male	38.18653	61.39062	3	UG

HYPOTHESIS TESTING

T - Test

A *t* test is a statistical test that is used to compare the means of two groups. It is often used in hypothesis testing to determine whether a process or treatment actually has an effect on the population of interest, or whether two groups are different from one another.
You want to know whether the mean petal length of iris flowers differs according to their species. You find two different species of irises growing in a garden and measure 25 petals of each species. You can test the difference between these two groups using a *t* test and null and alternative hypotheses.

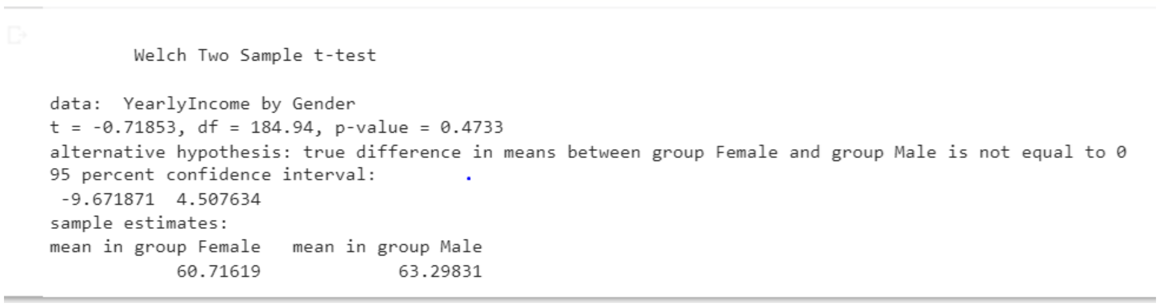
- The null hypothesis (H_0) is that the true difference between these group means is zero.
- The alternate hypothesis (H_a) is that the true difference is different from zero.

The formula for the two-sample *t* test (a.k.a. the Student's t-test) is shown below.

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{(s^2(\frac{1}{n_1} + \frac{1}{n_2}))}}$$

In this formula, *t* is the *t* value, x_1 and x_2 are the means of the two groups being compared, s_2 is the pooled standard error of the two groups, and n_1 and n_2 are the number of observations in each of the groups.

t.test(YearlyIncome ~ Gender, data = customer_data)

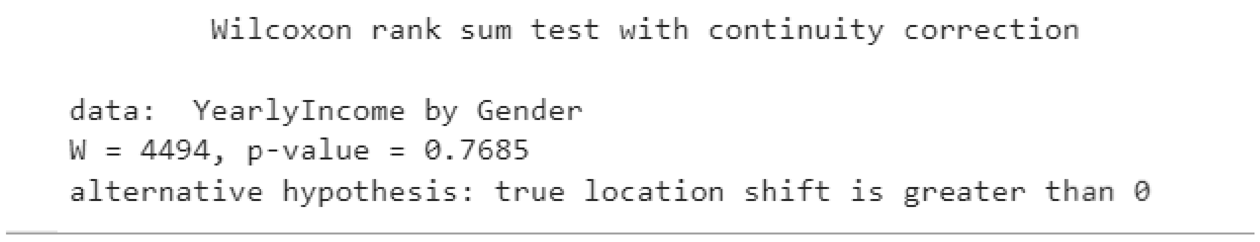


```
	Welch Two Sample t-test

data:  YearlyIncome by Gender
t = -0.71853, df = 184.94, p-value = 0.4733
alternative hypothesis: true difference in means between group Female and group Male is not equal to 0
95 percent confidence interval:
 -9.671871  4.507634
sample estimates:
mean in group Female   mean in group Male 
            60.71619             63.29831 
```

WILCOXON TEST

The Wilcoxon signed rank test is the non-parametric of the dependent samples t-test. Because the dependent samples t-test analyzes if the average difference of two repeated measures is zero, it requires metric (interval or ratio) and normally distributed data; the Wilcoxon sign test uses ranked or ordinal data; thus, it is a common alternative to the dependent samples t-test when its assumptions are not met. The Wilcoxon signed rank test relies on the *W*-statistic. For large samples with *n*>10 paired observations the *W*-statistic approximates a normal distribution. The *W* statistic is a non-parametric test, thus it does not need multivariate normality in the data.

```
wilcox.test(YearlyIncome ~ Gender, data = customer_data,alternative =
'greater')
```



```
        Wilcoxon rank sum test with continuity correction

data:  YearlyIncome by Gender
W = 4494, p-value = 0.7685
alternative hypothesis: true location shift is greater than 0
```

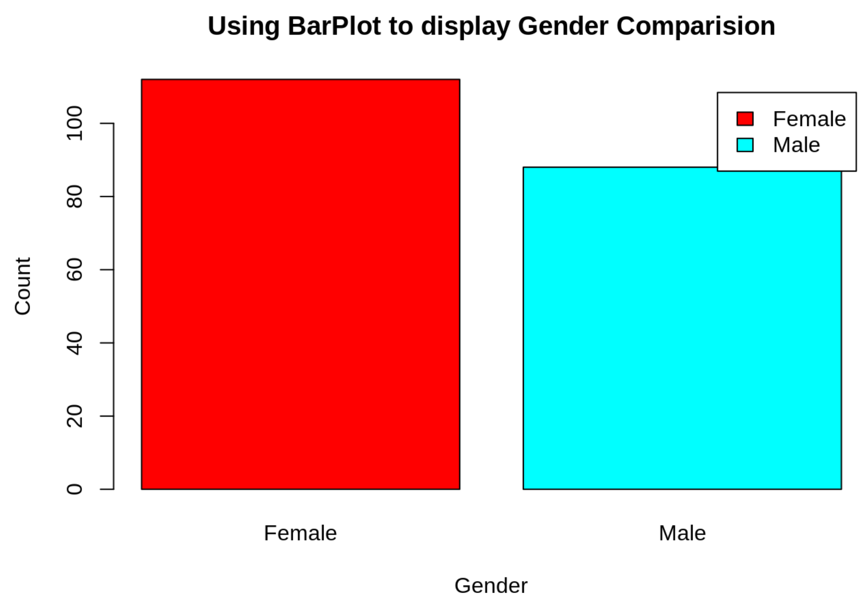
HYPOTHESIS TESTING CONCLUSION

1. We got the 2 attributes as important attributes
2. The first being the annual income
3. The second being the age
4. The analysis further should be based on these two attributes
5. Gender also shows importance in the hypothesis testing but the analysis will be done on annual income vs age

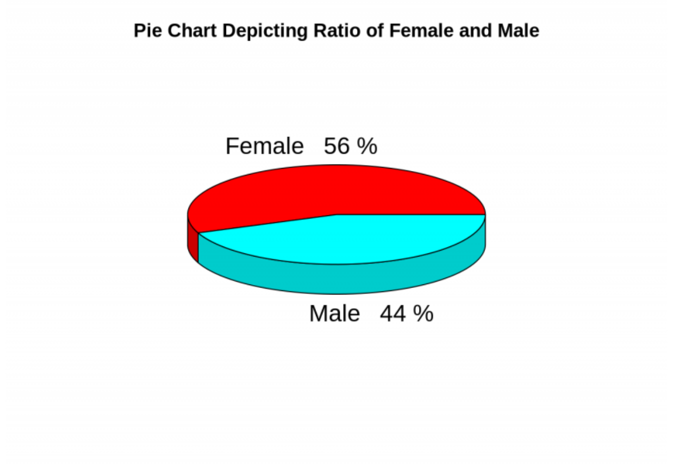
Customer Gender Visualization

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

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



```
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%.

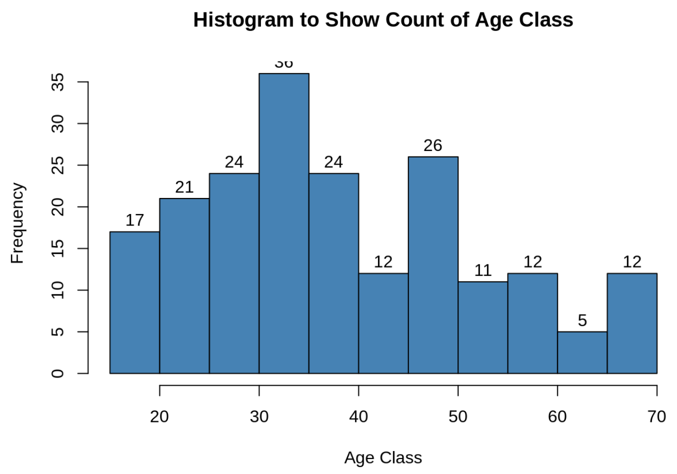
Visualization of Age Distribution

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.

```
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)
```



```r
boxplot(customer_data$Age,
        col="#ff0066",
        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.

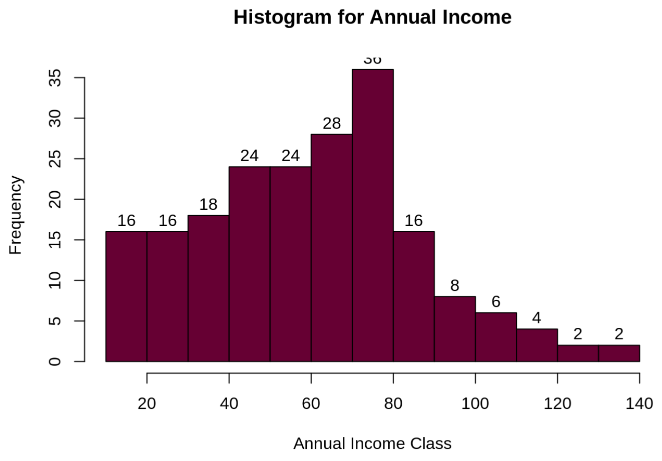
Analysis of the Annual Income of the Customers

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.

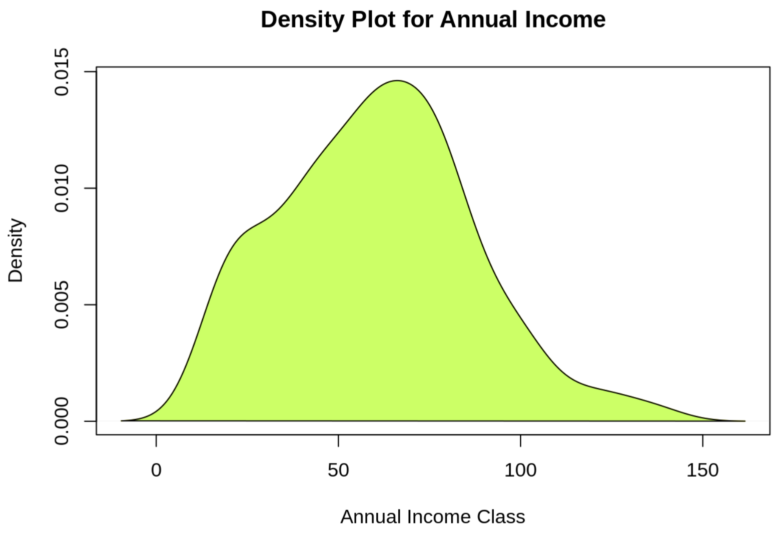
```r
summary(customer_data$Annual.Income..k..)
```

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

```r
hist(customer_data$Annual.Income..k..,
    col="#660033",
    main="Histogram for Annual Income",
    xlab="Annual Income Class",
    ylab="Frequency",
    labels=TRUE)
```



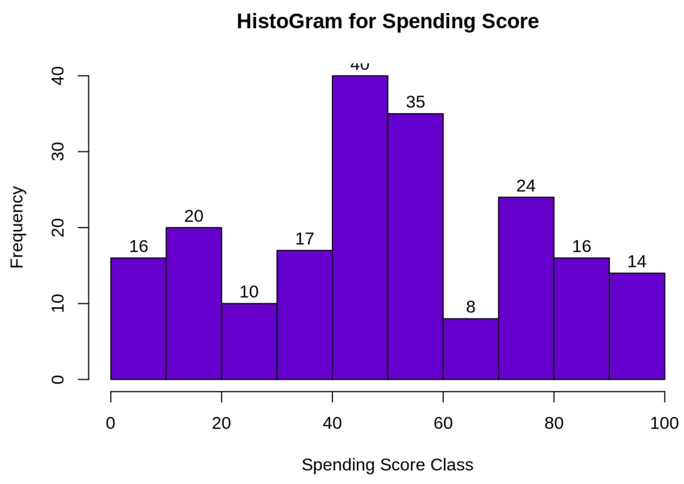
```
plot(density(customer_data$Annual.Income..k..),
    col="yellow",
    main="Density Plot for Annual Income",
    xlab="Annual Income Class",
    ylab="Density")
polygon(density(customer_data$Annual.Income..k..),
        col="#ccff66")
```



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

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



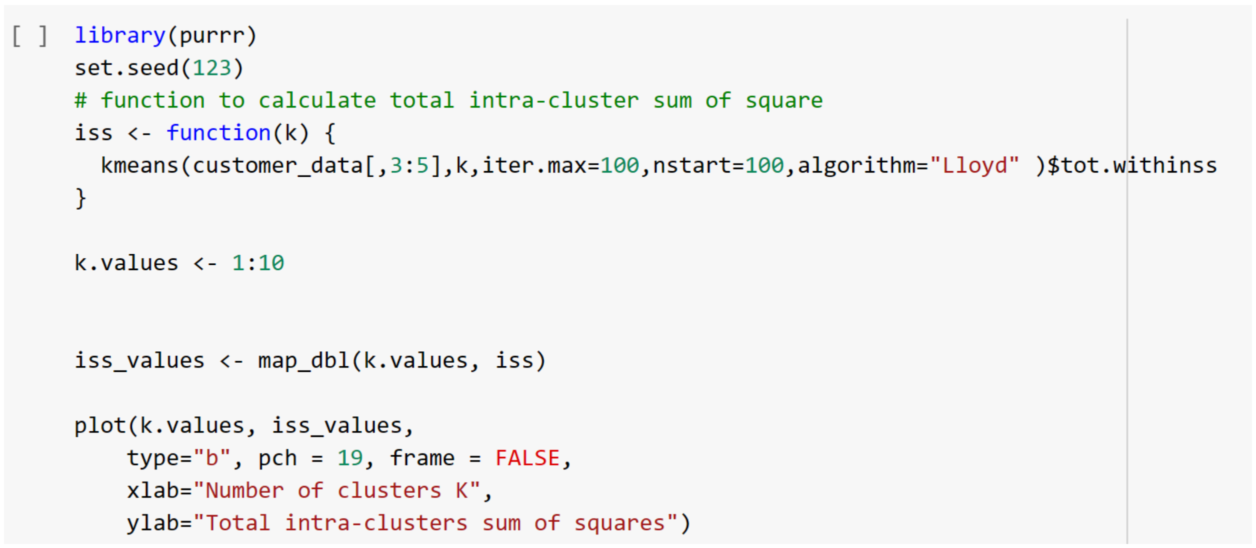
The minimum spending score is 1, maximum is 99 and the average is 50.20. We can see Descriptive Analysis of Spending Score is that Min is 1, Max is 99 and avg. is 50.20. From the histogram, we conclude that customers between class 40 and 50 have the highest spending score among all the classes.

MODEL SELECTION

MODEL 1- K-NN (K- Nearest Neighbour)

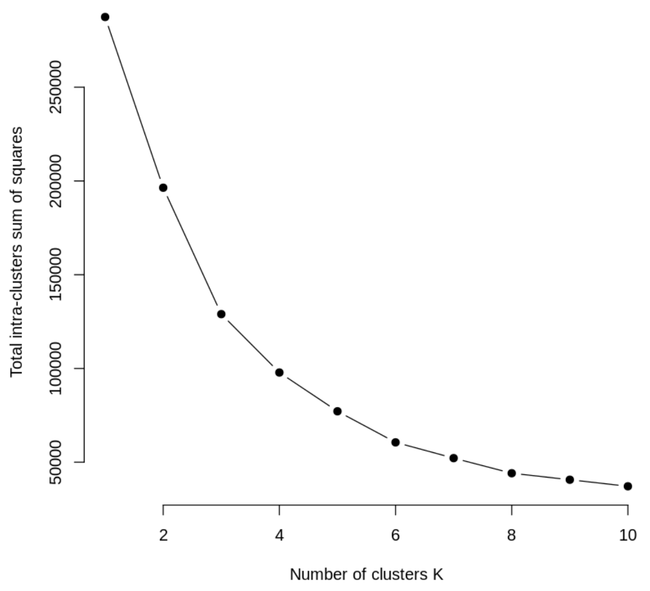
KNN is a supervised learning algorithm, meaning that the examples in the dataset must have labels assigned to them/their classes must be known. There are two other important things to know about KNN. First, KNN is a non-parametric algorithm. This means that no assumptions about the dataset are made when the model is used.
A KNN model calculates similarity using the distance between two points on a graph. The greater the distance between the points, the less similar they are. There are multiple ways of calculating the distance between points, but the most common distance metric is just Euclidean distance (the distance between two points in a straight line).

To find the K value to be used in the model:

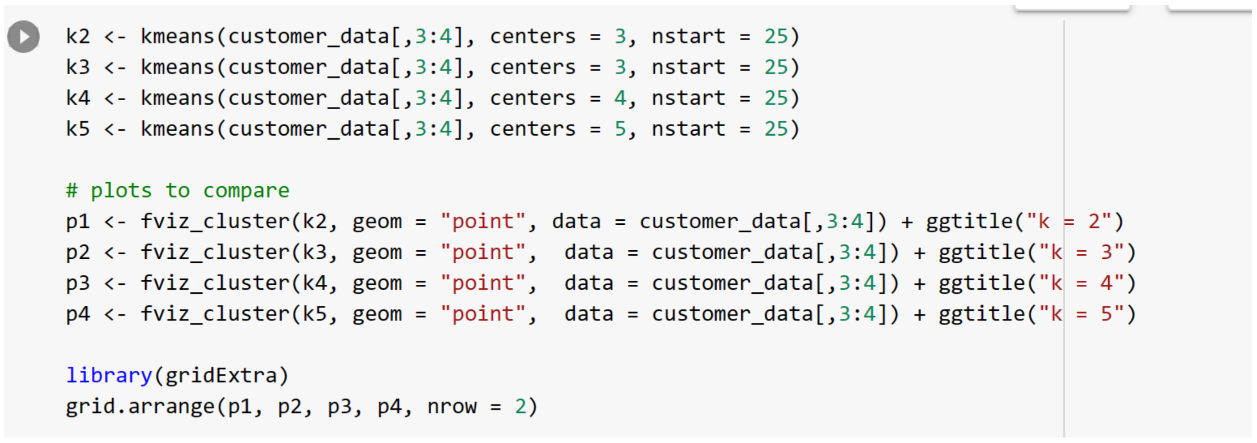


```r
library(purrr)
set.seed(123)
# function to calculate total intra-cluster sum of square
iss <- function(k) {
  kmeans(customer_data[,3:5],k,iter.max=100,nstart=100,algorithm="Lloyd" )$tot.withinss
}

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")
```

Using Elbow method to determine number of clusters(K):

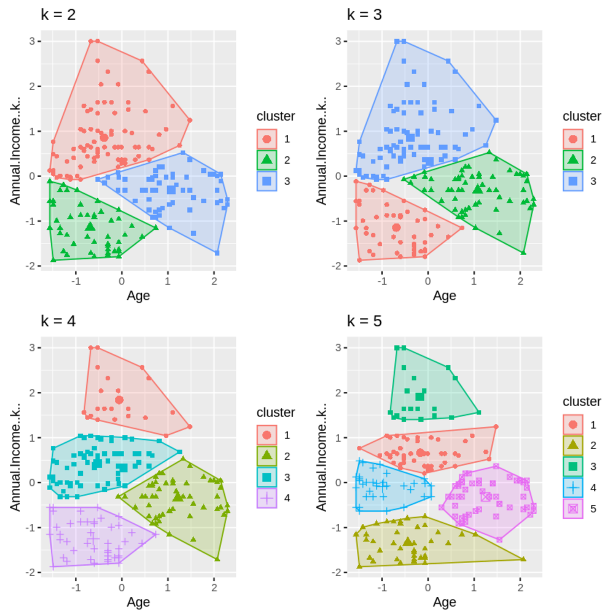


Using KNN we can have different clusters of the data to find the suitable cluster:



```
k2 <- kmeans(customer_data[,3:4], centers = 3, nstart = 25)
k3 <- kmeans(customer_data[,3:4], centers = 3, nstart = 25)
k4 <- kmeans(customer_data[,3:4], centers = 4, nstart = 25)
k5 <- kmeans(customer_data[,3:4], centers = 5, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = customer_data[,3:4]) + ggtitle("k = 2")
p2 <- fviz_cluster(k3, geom = "point",  data = customer_data[,3:4]) + ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = customer_data[,3:4]) + ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = customer_data[,3:4]) + ggtitle("k = 5")

library(gridExtra)
grid.arrange(p1, p2, p3, p4, nrow = 2)
```

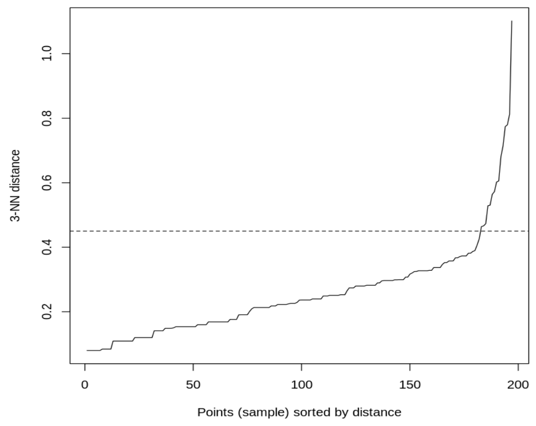
Random Clusters generated:



To measure Distance of the points in the clusters:

```
[ ]  eps_plot = kNNdistplot(customer_prep, k=3)

     eps_plot %>% abline(h = 0.45, lty = 2)
```

Inertia of the points in each clusters:



Model 2- DBScan

Clustering is an unsupervised learning technique used to group data based on similar characteristics when no pre-specified group labels exist. This technique is used for statistical data analysis across many fields.

DBSCAN has two parameters. The first is ε, epsilon ("esp"), which defines the maximum distance allowed between two points within the same cluster. The second is minimum samples ("MinPts"), which defines the minimum number of data points required to form a distinct cluster. So, MinPts is the minimum number of neighbors contained within a cluster with radius/max length of esp.

DBSCAN is more flexible when it comes to the size and shape of clusters than other partitioning methods, such as K-means. It is able to identify clusters that differ in size and shape from one another, which makes it more useful for messy, real life data.

Creating DBScan model:

```
[ ]  set.seed(50)

     # creation of an object km which store the output of the function kmeans
     d <- dbscan::dbscan(customer_prep, eps = 0.45, MinPts =  2)
     d
```

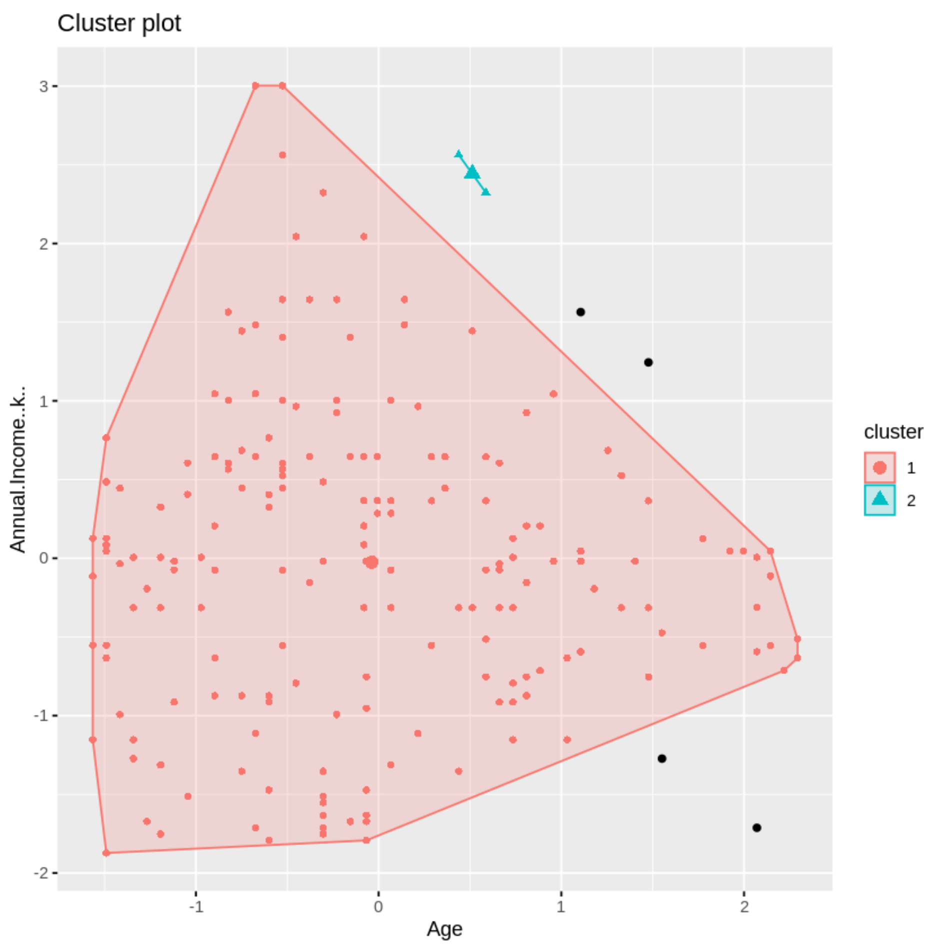
Clustering progress of DBScan model:

```
"converting argument MinPts (fpc) to minPts (dbscan)!"
DBSCAN clustering for 197 objects.
Parameters: eps = 0.45, minPts = 2
Using euclidean distances and borderpoints = TRUE
The clustering contains 2 cluster(s) and 4 noise points.

  0   1   2
  4 191   2

Available fields: cluster, eps, minPts, dist, borderPoints
```

Visualization of Cluster created by DBScan model:

```
[ ]  # cluster visualisation
     fviz_cluster(d, customer_prep, geom = "point")
```

Clusters created and outliers in Data removed from clusters:

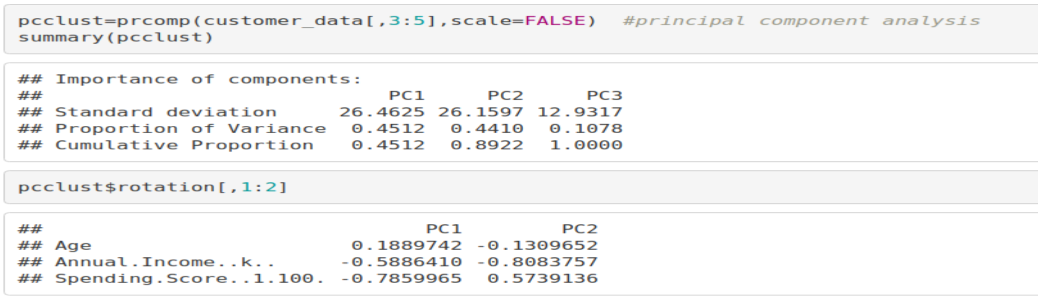


CONCLUSION

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

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

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



```
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
```

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.

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.