SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Project Review – 1

Instructions for First Review:

1. Project Title and Batch members Details.

Template:

Batch Size	Reg No	Name	Title of the Project
2	18MIS1042	SUDHEER KUMAR N	**Customer Segmentation using K-Means Algorithm**
	18MIS1010	AMITH SAJJA	

2. One page Abstract regarding your project.

In today's fast-moving world of marketing from product-orientation to customer-orientation, the management of customer treatment can be seen as a key to achieve revenue growth and profitability. Knowledge of customer behavior can help marketing managers re-evaluate their strategies with the customers and plan to improve and expand their application of the most effective strategies. B2B or business customers are more complex, their buying process is more complicated and their sales value is greater. The business marketers usually prefer to cooperate with fewer but larger buyers than the final consumer marketer. As a business transaction requires more decision makings and more professional buying effort than the consumer market does, the efficient relationship with business customers is of paramount importance. Most customer segmentation approaches based on customer value fail to account for

the factor of time and the trend of value changes in their analysis. The business done today runs on the basis of innovative ideas as there are large number of potential customers who are confounded to what to buy and what not to buy. The companies doing the business are also not able to diagnose the target potential customers. This is where the machine learning comes into picture, the various algorithms are applied to identify the hidden patterns in the data for better decision making. The concept of which customer segment to target is done using the customer segmentation process using the clustering technique. In this project, the clustering algorithm we are going to use is K-means algorithm which is the partitioning algorithm, to segment the customers according to the similar characteristics. To determine the optimal clusters, elbow method is used.

Abstract:

(i)   Title of the project.
        Customer Segmentation using K-Means Algorithm

(ii)   Brief explanation about your project (Not more that one page).

        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

behavioural patterns play a crucial role in determining the company direction towards addressing the various segments.

3. Concept of your project and modules in the project.
    (i)   List the Modules in your project
- ✓ Data Exploration
- ✓ Data Pre-processing
- ✓ Data Visualization
- ✓ Data Analysis

    (ii)   List the concepts to be applied in your project.
- ✓ Un-Spervised Algorithm(K-Means Clustering)
- ✓ Clustering
- ✓ Elbow Method
- ✓ Seaborn,matplotlib,ggplot2,etc.. – Visualizations
- ✓ All necessary concepts will be used for all the Modules involved in the project

    (iii)   Platform & Language to be used for Implementation.

    The project wil be implemented in both R and Python
- ✓ Python -Jupyter
- ✓ R-RStudio

Note:

    For the first review kindly do the above 3 steps and upload the same in Moodle

REVIEW-2

Objective:

1. To cluster customers based on common purchasing behaviors for future operations/marketing projects
2. To incorporate best mathematical, visual, programming, and business practices into a thoughtful analysis that is understood across a variety of contexts and disciplines
3. The goal of segmenting customers is to decide how to relate to customers in each segment in order to maximize the value of each customer to the business.

Methodology:

The data set used to implement clustering and Kmeans algorithm was collected from a store of shopping mall. The data set contains 5 attributes and has 200 tuples, representing the data of 200 customers. The attributes in the data set has CustomerId, gender, age, annual income(k$), spending score on the scale of (1-100).

Literature Review:

Customer Segmentation

Customer Segmentation Over the years, as there is very strong competition in the business world, the organizations have to enhance their profits and business by satisfying the demands of their customers and attract new customers according to their needs. The identification of customers and satisfying the demands of each customer is a very complex and tedious task. This is because customers may be different according to their demands, tastes, preferences and so on. Instead of "one-size-fits-all" approach, customer segmentation clusters the customers into groups sharing the same properties or behavioral characteristics. Customer segmentation is a strategy of dividing the market into homogenous groups. The data used in customer segmentation technique that divides the customers into groups depends on various factors like, data geographical conditions, economic conditions, demographical conditions as well as behavioral patterns. The customer segmentation technique allows the business to make better use of their marketing budgets, gain a competitive edge over their rival companies, demonstrating the better knowledge of the needs of the customer. It also helps an organization in, increasing their marketing efficiency, determining new market opportunities, making better brand strategy, identifying customers retention.

Clustering and K-Means Algorithm

Clustering algorithms generates clusters such that within the clusters are similar based on some characteristics. Similarity is defined in terms of how close the objects are in space. K-means algorithm in one of the most popular centroid based algorithms. Suppose data set, D, contains n objects in space. Partitioning methods distribute the objects in D into k clusters, $C_1,...,C_k$, that is, $C_i \subset D$ and $C_i \cap C_j = \emptyset$ for $(1 \leq i, j \leq k)$. A centroid-based partitioning technique uses the centroid of a cluster, C_i, to represent that cluster. Conceptually, the centroid of a cluster is its center point. The difference between an object $p \in C_i$ and c_i, the representative of the cluster, is measured by dist(p,ci), where dist(x,y) is the Euclidean distance between two points x and y.

Algorithm

The k-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster. Input: k: the number of clusters, D: a data set containing n objects. Output: A set of k clusters. Method: (1) arbitrarily choose k objects from D as the initial cluster centers; (2) repeat (3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster; (4) update the cluster means, that is, calculate the mean value of the objects for each cluster; (5) until no change.

Implementation of a Customer Segmentation in R

In the first step of this Fundamental Data Analytics 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.

Code

```
customer_data <- read.csv("Mall_Customers.csv")
str(customer_data)
#Column Names
 names(customer_data)
```

Output

```r
customer_data <- read.csv("Mall_Customers.csv")
str(customer_data)
#Column Names
names(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 ...
 [1] "CustomerID"             "Genre"
 [3] "Age"                    "Annual.Income..k.."
 [5] "Spending.Score..1.100."
```

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

Code

head(customer_data)

```r
summary(customer_data$Age)
```

```r
sd(customer_data$Age)
```

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

Output

```r
summary(customer_data$Spending.Score..1.100.)
sd(customer_data$Spending.Score..1.100.)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00   34.75   50.00   50.20   73.00   99.00
 [1] 25.82352
```

Output

```
head(customer_data)
```

Description: df [6 x 5]

	CustomerID <int>	Genre <chr>	Age <int>	Annual.Income..k.. <int>
1	1	Male	19	15
2	2	Male	21	15
3	3	Female	20	16
4	4	Female	23	16
5	5	Female	31	17
6	6	Female	22	17

6 rows | 1-5 of 5 columns

```
summary(customer_data$Age)
```

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

```
sd(customer_data$Age)
```

```
[1] 13.96901
```

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

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  15.00   41.50   61.50   60.56   78.00  137.00
 [1] 26.26472
```

```
summary(customer_data$Spending.Score..1.100.)
sd(customer_data$Spending.Score..1.100.)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00   34.75   50.00   50.20   73.00   99.00
 [1] 25.82352
```

Customer Gender Visualization

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

Code

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

Output

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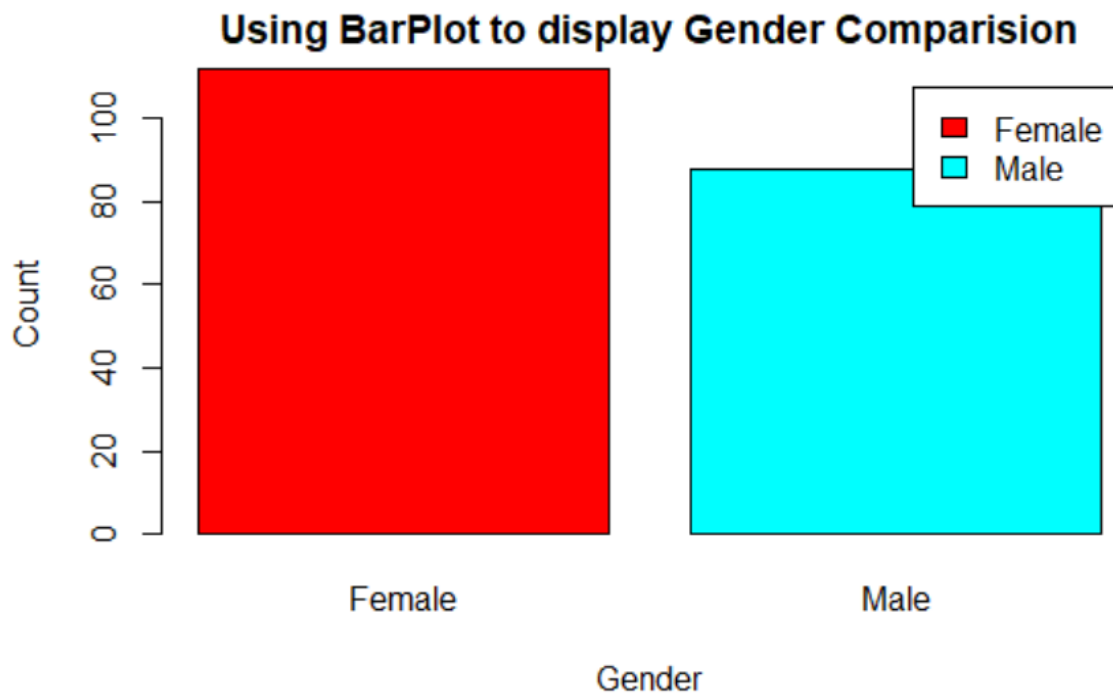

Inference:- From the above bar plot, we observe that the number of females is higher than the males.

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

Code
```
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")
```

Output
```
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")
```


Pie Chart Depicting Ratio of Female and Male

Female 56 %

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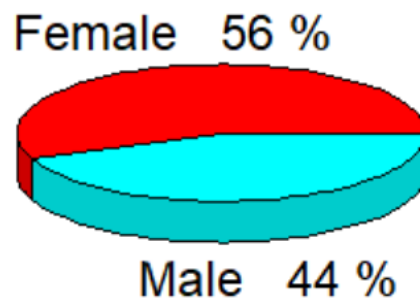
Inference: From the above graph, we can conclude that the percentage of females is **56%**, whereas the percentage of male in the customer dataset is **44%.**

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.

Code

```
summary(customer_data$Age)
```

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

Output

```
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)
```


Histogram to Show Count of Age Class
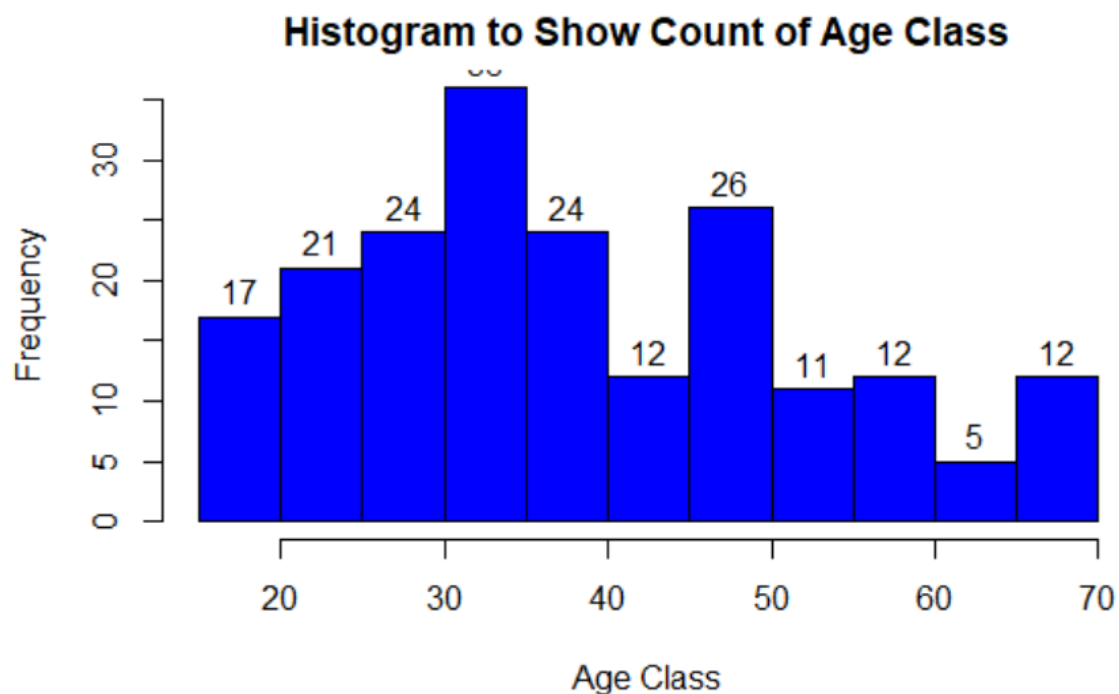
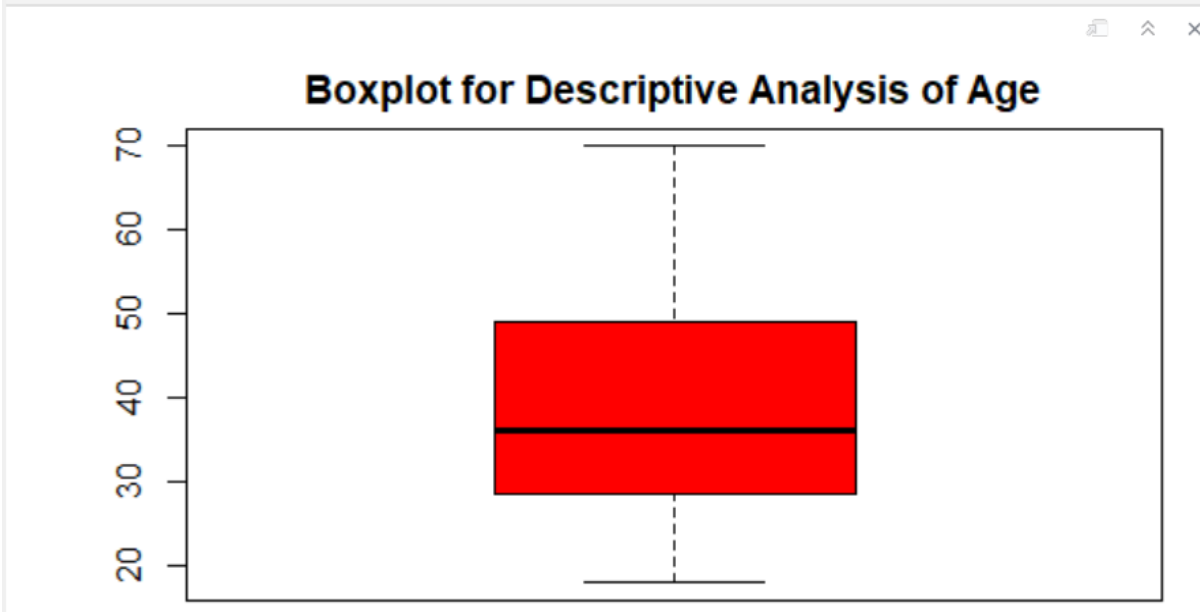**Code**

```
boxplot(customer_data$Age,
       col="red",
```

```
        main="Boxplot for Descriptive Analysis of Age")
```

Output

```
boxplot(customer_data$Age,
        col="red",
        main="Boxplot for Descriptive Analysis of Age")|
```

⤢ ⌃ ✕

Boxplot for Descriptive Analysis of Age

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

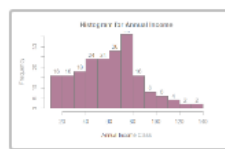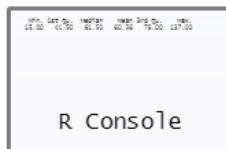## Analysis of the Annual Income of the Customers

we will create visualizations to analyse the annual income of the customers. We will plot a histogram and then we will proceed to examine this data using a density plot.
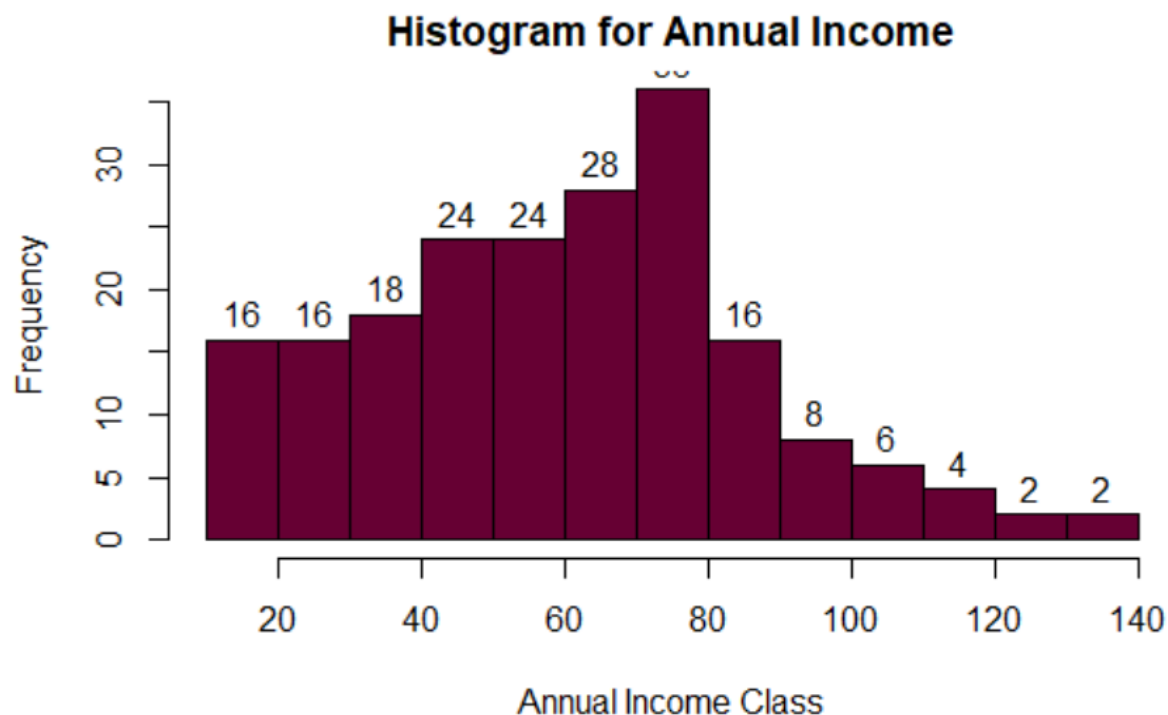
Code

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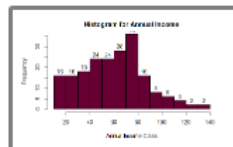
Output

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

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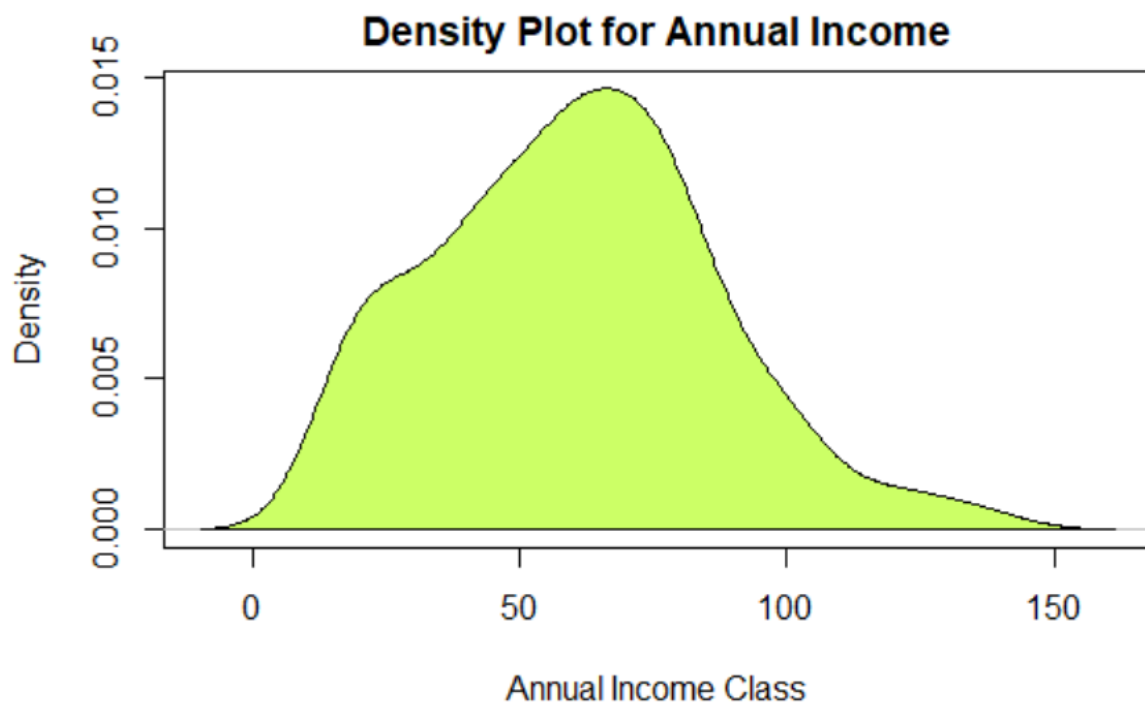

Histogram for Annual Income

Code

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")
```

Output

```
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")
```


Inference: 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

Code

```
summary(customer_data$Spending.Score..1.100.)

boxplot(customer_data$Spending.Score..1.100.,
    horizontal=TRUE,
    col="#990000",
    main="BoxPlot for Descriptive Analysis of Spending Score")
```

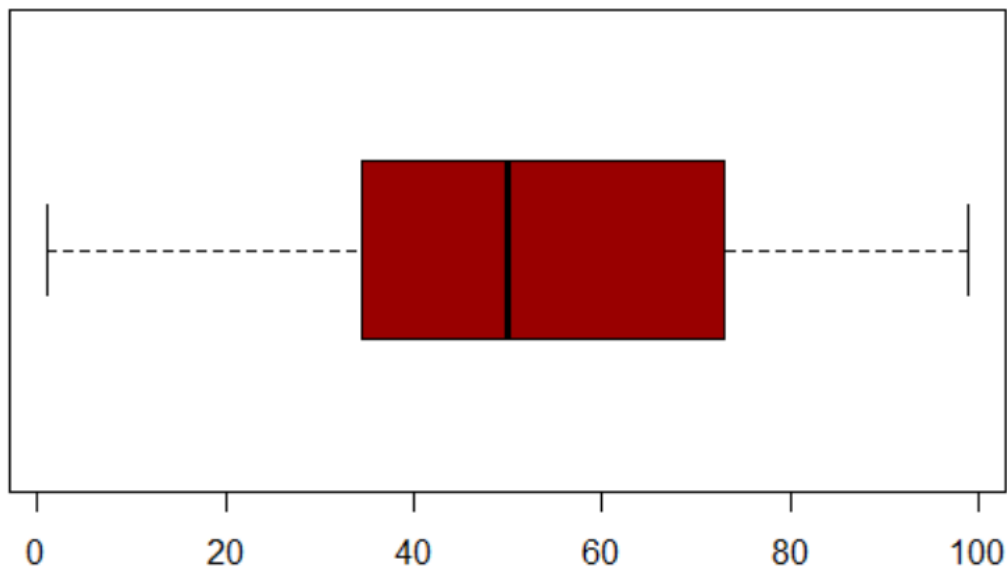## Output

```
summary(customer_data$Spending.Score..1.100.)
boxplot(customer_data$Spending.Score..1.100.,
    horizontal=TRUE,
    col="#990000",
    main="BoxPlot for Descriptive Analysis of Spending Score")
```

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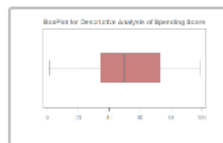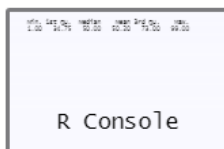
```
    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    1.00   34.75   50.00   50.20   73.00   99.00
```

BoxPlot for Descriptive Analysis of Spending Score

Code

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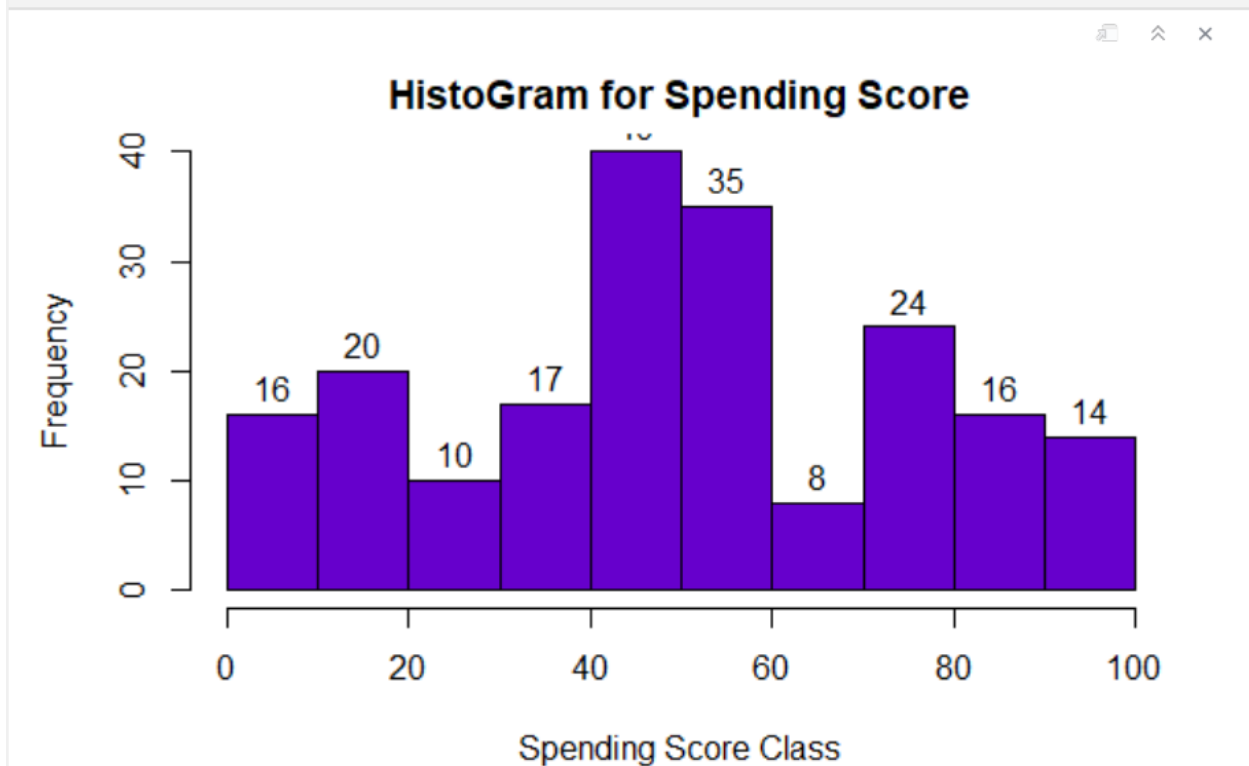
Output

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


HistoGram for Spending Score

Inference: 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, we need to specify the number of clusters to use. We would like to utilize the optimal number of clusters. There are three popular methods to determine the optimal clusters –

1. Elbow method
2. Silhouette method
3. Gap statistic

1. 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.

Code

```
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")
```

Output

```
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")
```
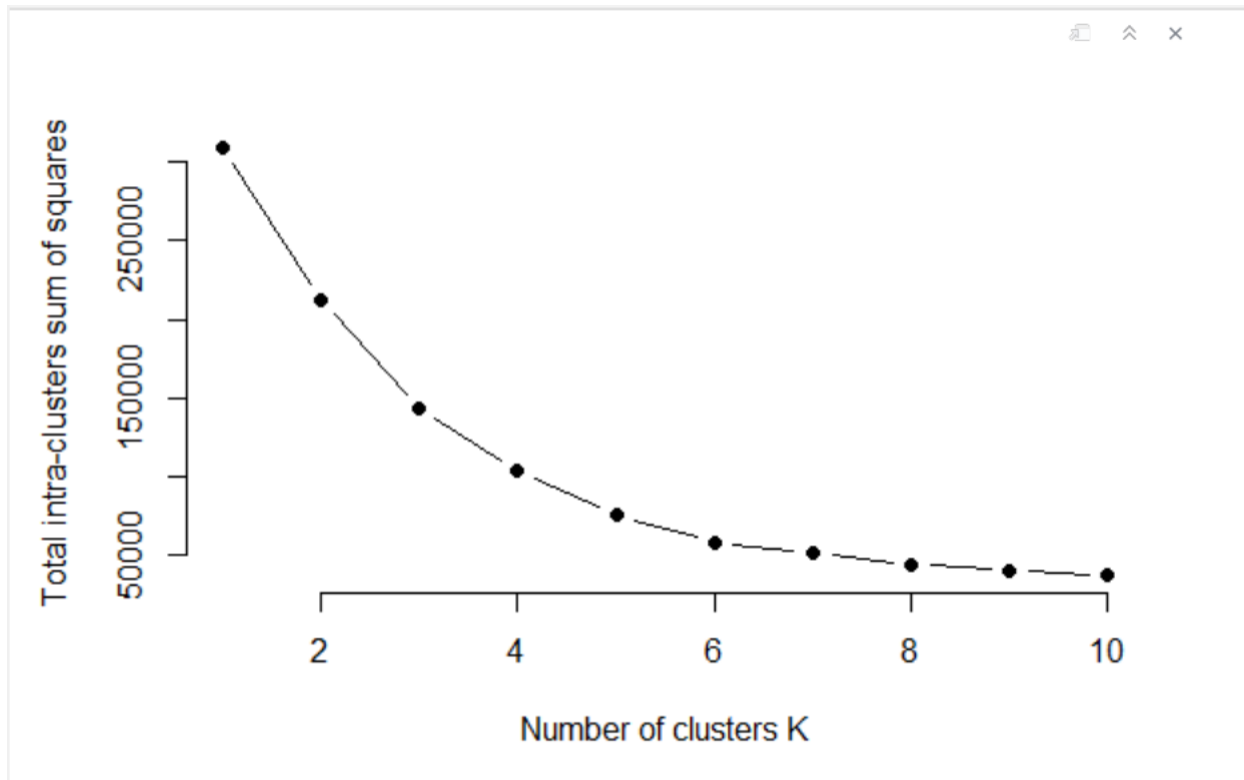
Inference: From the above graph, we can conclude that 4 is the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot.

2. 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.

Code

```
library(cluster)
library(gridExtra)
library(grid)
```
When n=2

```
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")))

```
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")))
```

Silhouette plot of (x = k2$cluster, dist = dist(customer_data[, 3:5],

n = 200

2 clusters C_j

$j : n_j | ave_{i \in Cj}$ s_i

1 : 85 | 0.31

2 : 115 | 0.28

| 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |

Silhouette width s_i

Average silhouette width : 0.29

When n=3

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")))

```
{r pressure, echo=TRUE}
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")))
```

Silhouette plot of (x = k3$cluster, dist = dist(customer_data[, 3:5],

n = 200

3 clusters C_j

$j : n_j | ave_{i \in Cj} \quad s_i$

1 : 123 | 0.28

2 : 38 | 0.50

3 : 39 | 0.60

0.0   0.2   0.4   0.6   0.8   1.0

Silhouette width s_i

Average silhouette width : 0.38

When n=4

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")))

```
k3<-kmeans(customer_data[,3:5],4,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

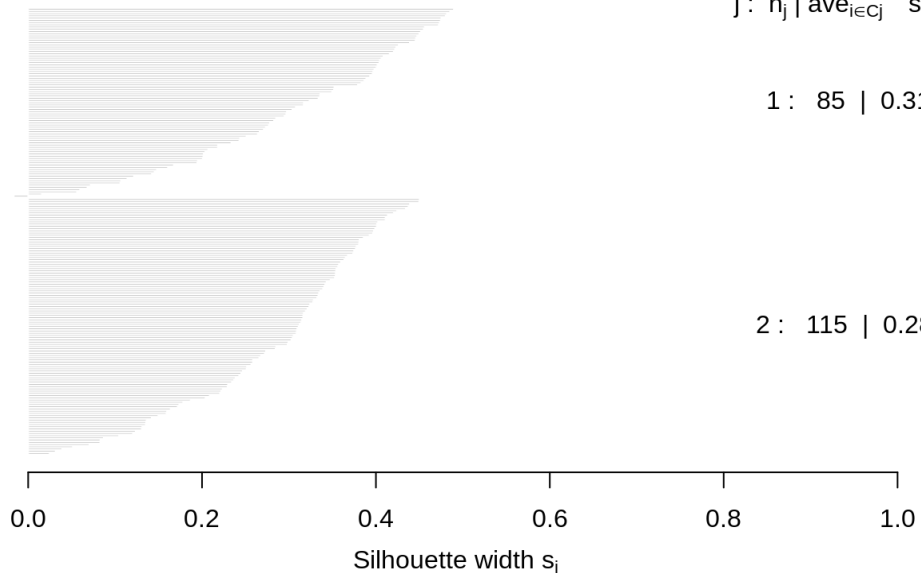**Silhouette plot of (x = k4$cluster, dist = dist(customer_data[, 3:5],**

n = 200

4 clusters C_j

$j : n_j | ave_{i \in Cj} \quad s_i$

1 : 28 | 0.51

2 : 39 | 0.58

3 : 95 | 0.29

4 : 38 | 0.44

0.0   0.2   0.4   0.6   0.8   1.0

Silhouette width s_i

Average silhouette width : 0.41

When n=5

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")))

```
k3<-kmeans(customer_data[,3:5],5,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```
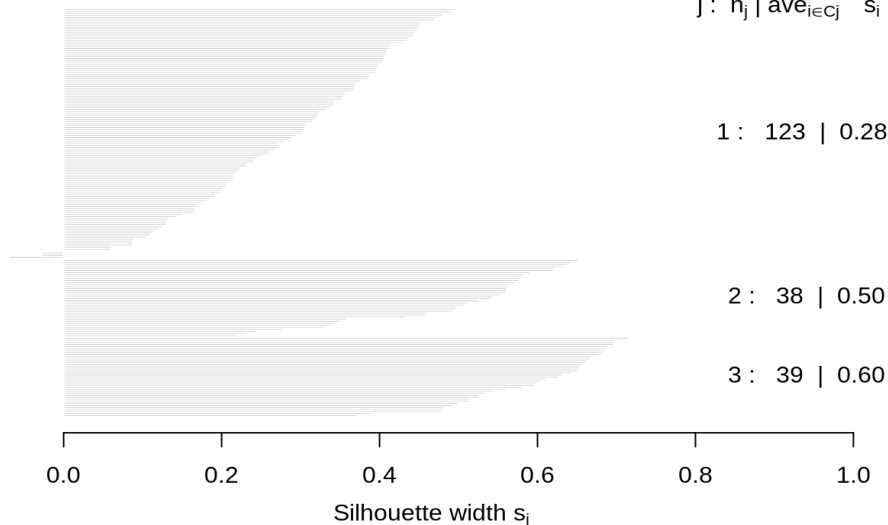
Silhouette plot of (x = k5$cluster, dist = dist(customer_data[, 3:5],

n = 200

	5 clusters C_j
	j : n_j | $ave_{i \in Cj}$ s_i
	1 : 23 | 0.42
	2 : 39 | 0.53
	3 : 23 | 0.60
	4 : 36 | 0.43
	5 : 79 | 0.37

0.0   0.2   0.4   0.6   0.8   1.0

Silhouette width s_i

Average silhouette width : 0.44

When n=6

k6<-kmeans(customer_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
s6<-plot(silhouette(k6$cluster,dist(customer_data[,3:5],"euclidean")))
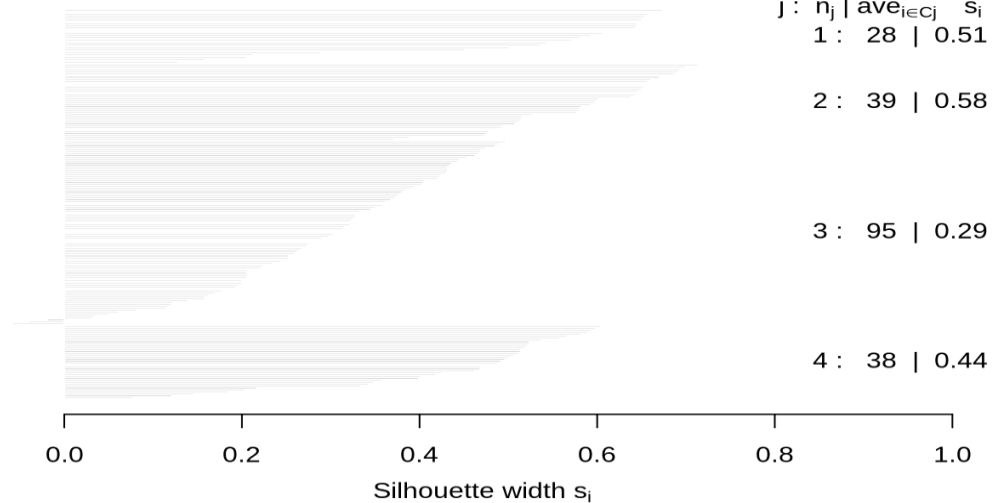
```
k3<-kmeans(customer_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

Silhouette plot of (x = k6$cluster, dist = dist(customer_data[, 3:5],

n = 200

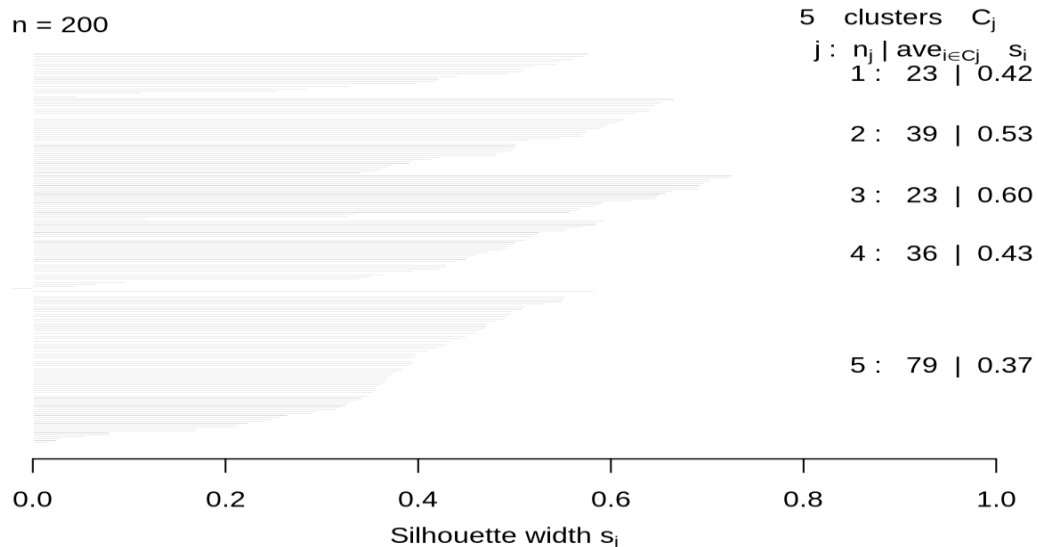| 6 | clusters | C_j |
	$j: n_j \mid ave_{i \in Cj}$	s_i
1:	39 |	0.50
2:	45 |	0.44
3:	21 |	0.42
4:	35 |	0.41
5:	22 |	0.58
6:	38 |	0.39

Silhouette width s_i

Average silhouette width : 0.45

When n=7

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")))

```
k3<-kmeans(customer_data[,3:5],7,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```
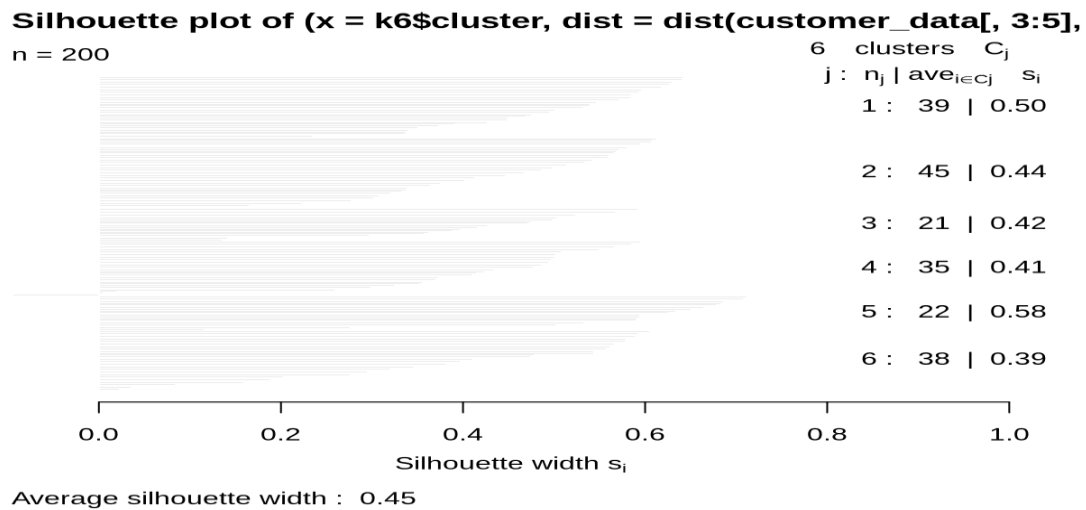
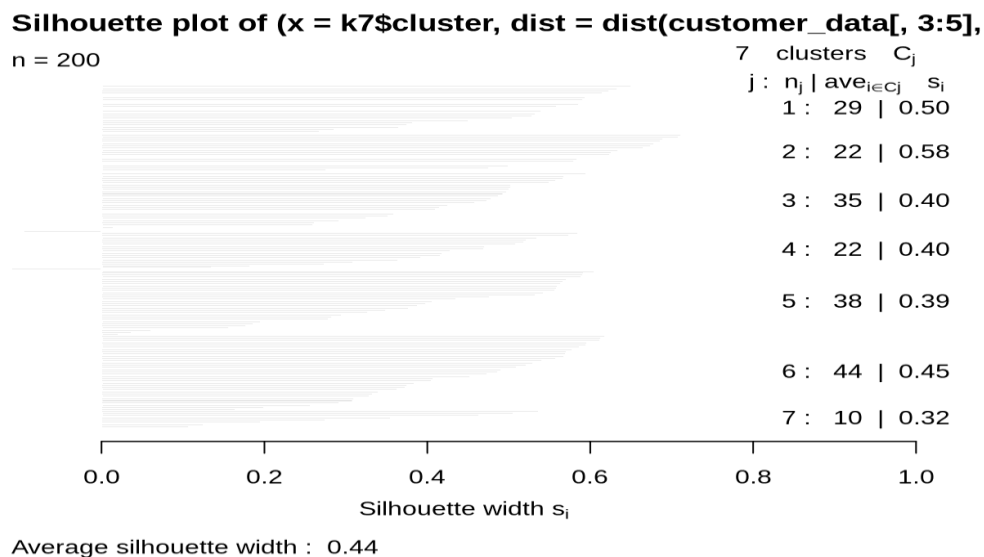**Silhouette plot of (x = k7$cluster, dist = dist(customer_data[, 3:5],**

n = 200

7	clusters	C_j
	$j: n_j \mid ave_{i \in Cj}$	s_i
1:	29 |	0.50
2:	22 |	0.58
3:	35 |	0.40
4:	22 |	0.40
5:	38 |	0.39
6:	44 |	0.45
7:	10 |	0.32

Silhouette width s_i

Average silhouette width : 0.44

When n=8

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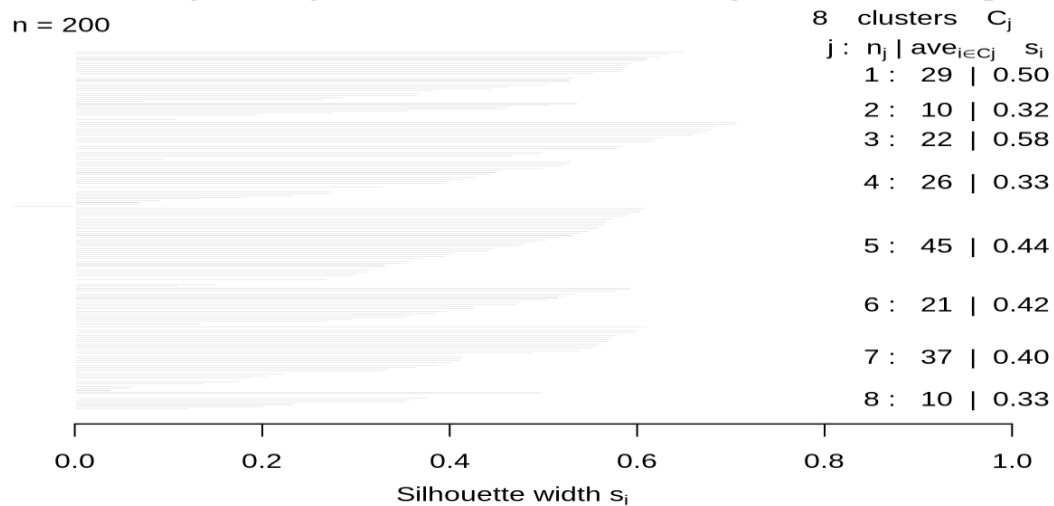s8<-plot(silhouette(k8$cluster,dist(customer_data[,3:5],"euclidean")))

```
k3<-kmeans(customer_data[,3:5],8,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

Silhouette plot of (x = k8$cluster, dist = dist(customer_data[, 3:5],

n = 200

8 clusters C_j

$j: n_j | ave_{i \in Cj}$ s_i

1 : 29 | 0.50
2 : 10 | 0.32
3 : 22 | 0.58
4 : 26 | 0.33
5 : 45 | 0.44
6 : 21 | 0.42
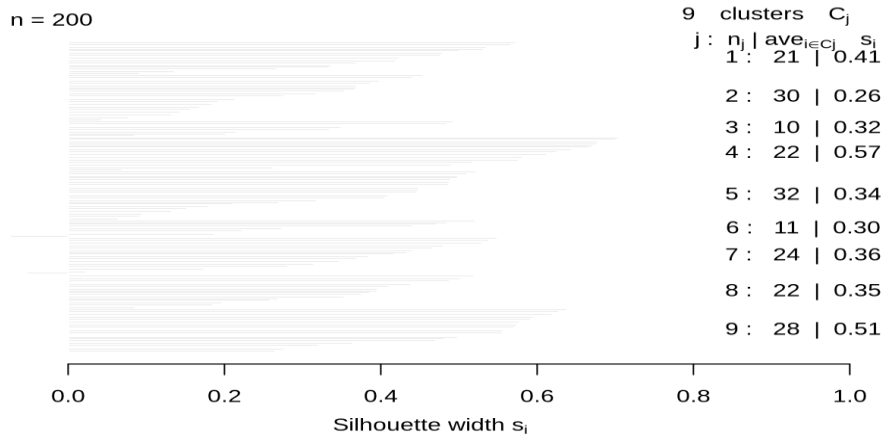7 : 37 | 0.40
8 : 10 | 0.33

Silhouette width s_i

Average silhouette width : 0.43

When n=9

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")))

```
k3<-kmeans(customer_data[,3:5],9,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

Silhouette plot of (x = k9$cluster, dist = dist(customer_data[, 3:5],

n = 200

9 clusters C_j

$j: n_j | ave_{i \in Cj}$ s_i

1 : 21 | 0.41
2 : 30 | 0.26
3 : 10 | 0.32
4 : 22 | 0.57
5 : 32 | 0.34
6 : 11 | 0.30
7 : 24 | 0.36
8 : 22 | 0.35
9 : 28 | 0.51

Silhouette width s_i

Average silhouette width : 0.39

When n=10

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")))

```
k3<-kmeans(customer_data[,3:5],10,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```
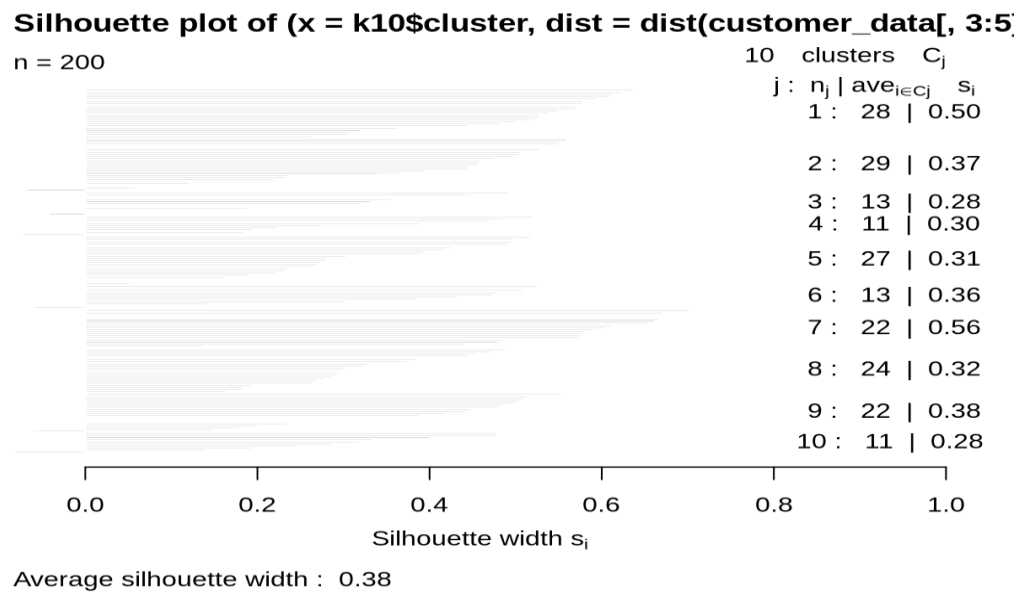
Silhouette plot of (x = k10$cluster, dist = dist(customer_data[, 3:5]

n = 200

10 clusters C_j

$j : n_j | ave_{i \in Cj}$ s_i

1 :	28	0.50
2 :	29	0.37
3 :	13	0.28
4 :	11	0.30
5 :	27	0.31
6 :	13	0.36
7 :	22	0.56
8 :	24	0.32
9 :	22	0.38
10 :	11	0.28

0.0   0.2   0.4   0.6   0.8   1.0

Silhouette width s_i

Average silhouette width : 0.38

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

Code

library(NbClust)
library(factoextra)

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

Output

```
library(NbClust)
library(factoextra)
fviz_nbclust(customer_data[,3:5], kmeans, method = "silhouette")
```

Optimal number of clusters