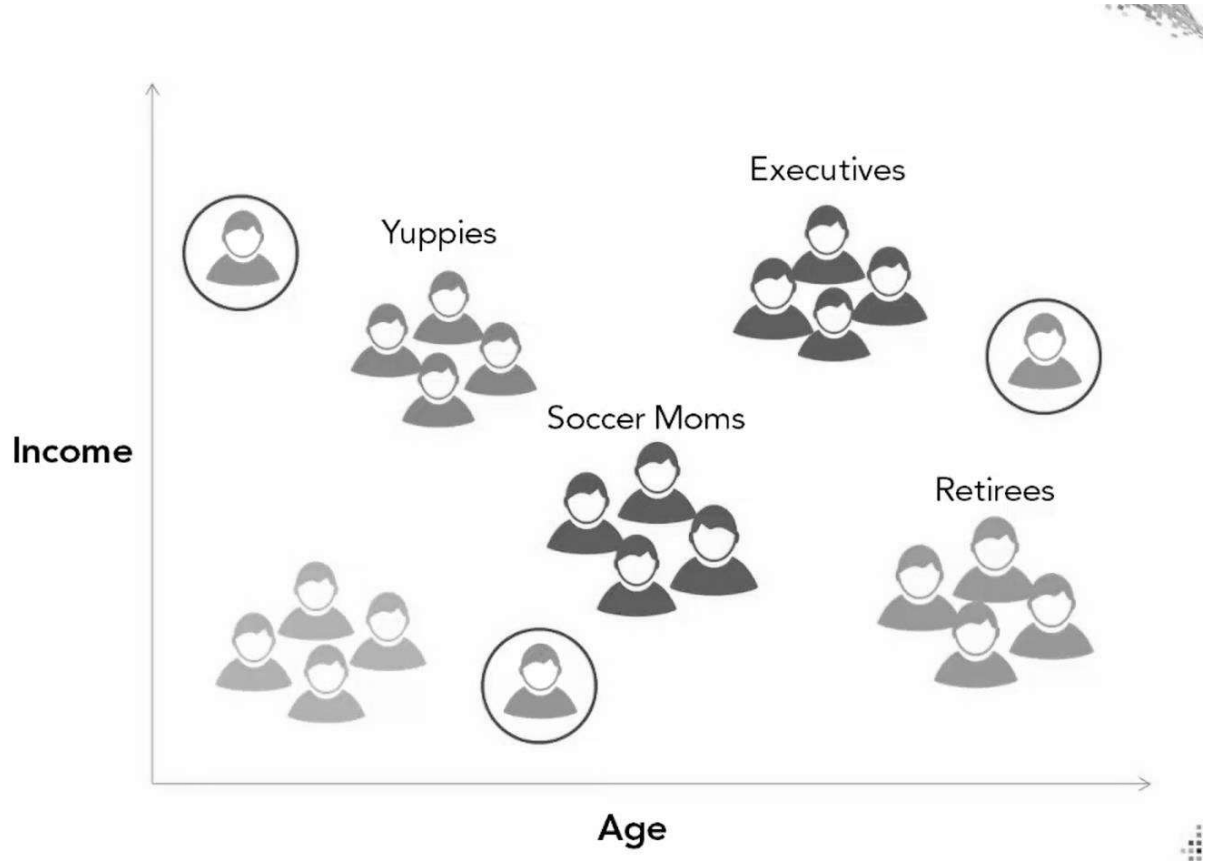
INTRODUCTION

WHAT IS CUSTOMER SEGMENTATION?



Customer segmentation is the process of dividing a company's customers into groups based on common characteristics so companies can market to each group effectively and appropriately.

In business-to-business marketing, a company might segment customers based on a wide range of factors, including:

- Industry
- Number of employees
- Products previously purchased from the company

- Location

In business-to-consumer marketing, companies often segment customers according to demographics that include:

- Age
- Gender
- Marital status
- Location (urban, suburban, rural)
- Life stage (single, married, divorced, empty-nester, retired, etc.)

Customer segmentation stands out as a pivotal application within the realm of unsupervised learning. By employing clustering methodologies, organizations can delineate various customer segments, enabling them to effectively target potential user bases. This machine learning endeavor specifically harnesses k-means clustering, a cornerstone algorithm tailored for clustering unlabeled datasets.

Exploration:

Customer segmentation uses the concept of k-means clustering, a technique that partitions data into distinct clusters based on shared characteristics. This iterative algorithm iteratively refines cluster centroids, minimizing within-cluster variance and thereby grouping similar data points together.

Practical Implementation:

In practical terms, k-means clustering allows businesses to categorize their customer base into similar groups, drawing insights from demographics, purchasing behaviours and preferences. This enables tailored marketing strategies and personalized customer experiences.

Advantages:

The utilization of k-means clustering in customer segmentation furnishes businesses with several advantages. It facilitates targeted marketing efforts, leading to heightened customer engagement and enhanced conversion rates. Moreover, by discerning the diverse needs of different customer segments, companies can optimize resource allocation and streamline operational efficiency.

In essence, customer segmentation via k-means clustering serves as a potent tool for businesses, empowering them to glean profound insights into their customer base and make informed, data-driven decisions. By harnessing this methodology, organizations can unlock new avenues for growth, foster customer loyalty, and maintain a competitive edge in today's dynamic marketplace.

SCOPE

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

PLATFORM: R STUDIO

R was purposefully crafted for statistical analysis, making it a prime choice for data science endeavors. While mastering R programming may pose challenges, especially for novices, the plethora of text analysis tools available in R streamline the process with straightforward commands. Contributing to R's popularity is its expansive library of packages, maintained by a thriving user community and housed on the Comprehensive R Archive Network (CRAN). These packages extend R's capabilities, with each one accompanied by comprehensive documentation and examples.

Text analysis, in particular, thrives in R, boasting a rich assortment of packages catering to various text processing needs, from basic string operations to sophisticated modeling techniques like Latent Dirichlet Allocation. R's flexibility shines through its ability to seamlessly switch between packages or integrate them, fostering interoperability and empowering users with diverse options. By grasping the fundamentals of text analysis in R, one gains access to a wealth of advanced features, exemplifying its versatility and utility in data science applications.

PROJECT SPECIFICATION

USED VERSION INFORMATION:

RStudio 2023.12.1+402 "Ocean Storm" Release (4da58325ffcff29d157d9264087d4b1ab27f7204, 2024-01-28) for windows

Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) RStudio/2023.12.1+402 Chrome/116.0.5845.190 Electron/26.2.4 Safari/537.36

DATASET USED:

- Mall_Customers.csv

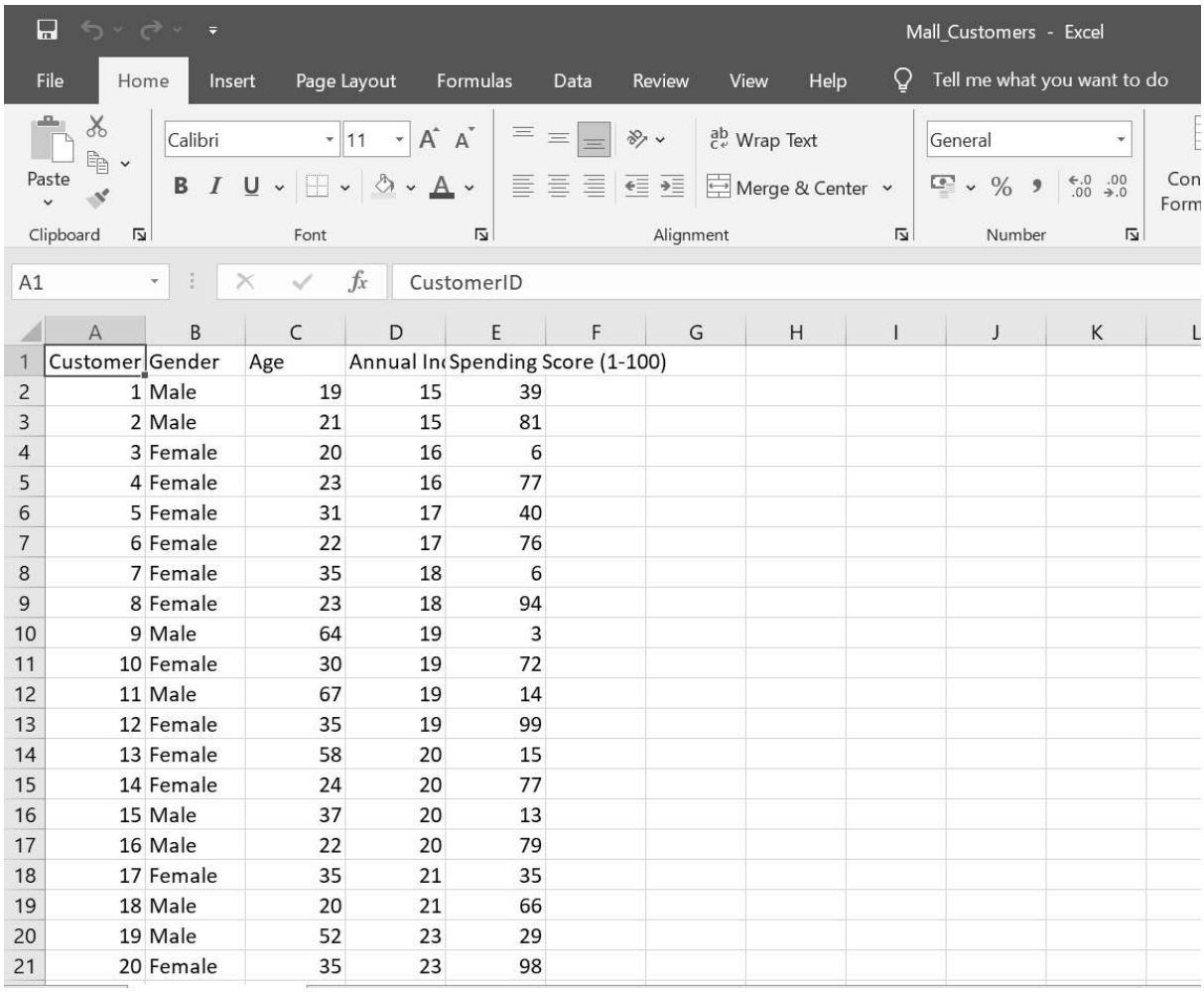
PACKAGES REQURIED:

- cluster
- gridExtra
- grid
- nbClust
- factoextra
- ggplot2
- dplyr
- plotrix

IMPLEMENTATION:

The first step is to read the data for performing analysis on. The data is saved in dataset named as Mall_Customers.csv. This dataset contains 400 record of various type of customers. The events saved in dataset are unstructured. To perform analysis, reading of data set is done using command “read.csv”.

customer_data=read.csv("C:\\Users\\vedan\\OneDrive\\Desktop\\Rda2\\Mall_Customers.csv")



	A	B	C	D	E	F	G	H	I	J	K	L
1	Customer	Gender	Age	Annual In	Spending Score (1-100)							
2	1	Male	19	15	39							
3	2	Male	21	15	81							
4	3	Female	20	16	6							
5	4	Female	23	16	77							
6	5	Female	31	17	40							
7	6	Female	22	17	76							
8	7	Female	35	18	6							
9	8	Female	23	18	94							
10	9	Male	64	19	3							
11	10	Female	30	19	72							
12	11	Male	67	19	14							
13	12	Female	35	19	99							
14	13	Female	58	20	15							
15	14	Female	24	20	77							
16	15	Male	37	20	13							
17	16	Male	22	20	79							
18	17	Female	35	21	35							
19	18	Male	20	21	66							
20	19	Male	52	23	29							
21	20	Female	35	23	98							

Structure of the dataset

```
> str(customer_data)
'data.frame':   400 obs. of  5 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 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 ...
```

Head of data

```
> head(customer_data)
  CustomerID Gender 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
```

Column wise Summary

Age

```
> 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] 13.95149
```

Annual Income

```
> 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] 26.23179
```

Spending Score

```
> summary(customer_data$Spending.Score..1.100.)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00   34.75   50.00   50.20   73.00   99.00
> sd(customer_data$Spending.Score..1.100.)
[1] 25.79114
```

Checking for null values in any of the columns

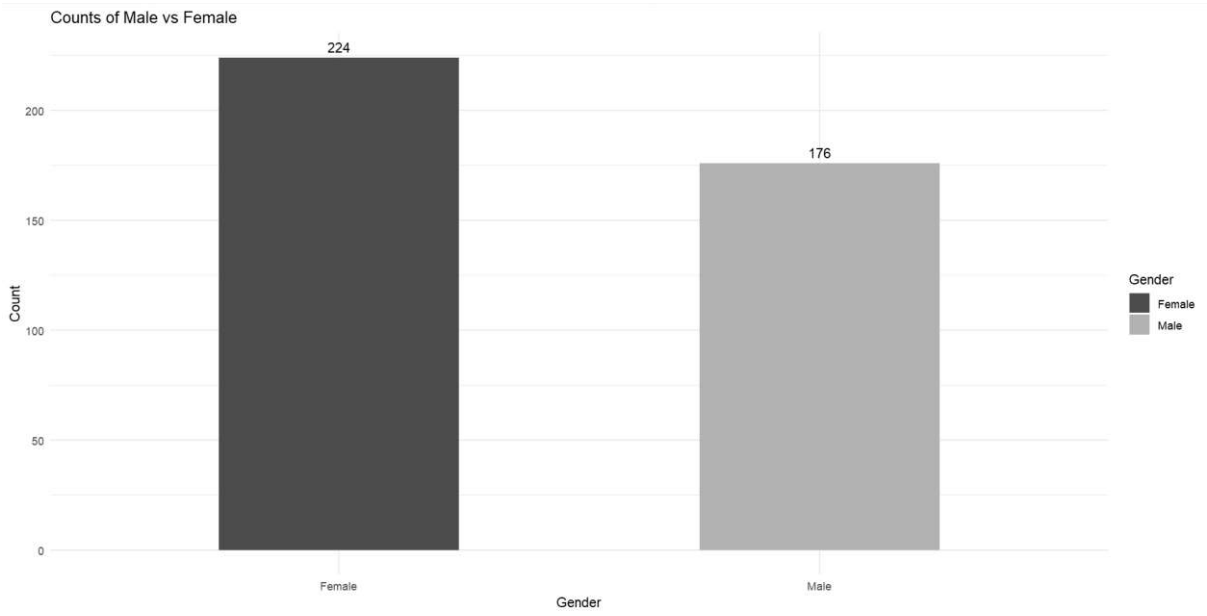
```
> print(paste("The number of null values: ", sum(is.na(customer_data))))
[1] "The number of null values:  0"
```

Visualization of data splits to see the distribution of each data and to find outliers:

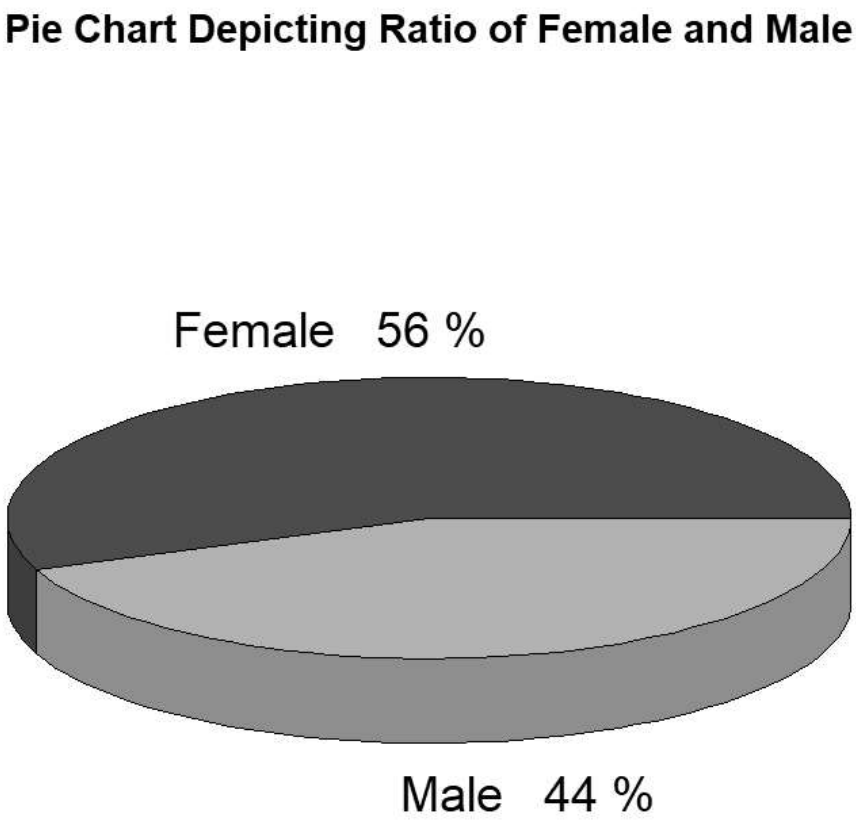
Gender

R code snippet:

```
# Visualization of Gender Distribution
a <- table(customer_data$Gender)
ggplot(customer_data, aes(x = Gender)) +
  geom_bar(aes(fill = Gender), width = 0.5) +
  geom_text(stat='count', aes(label=..count.., y=..count..), vjust=-0.5) +
  scale_fill_manual(values = rainbow(2)) +
  labs(title = "Counts of Male vs Female", x = "Gender", y = "Count") +
  theme_minimal()
```



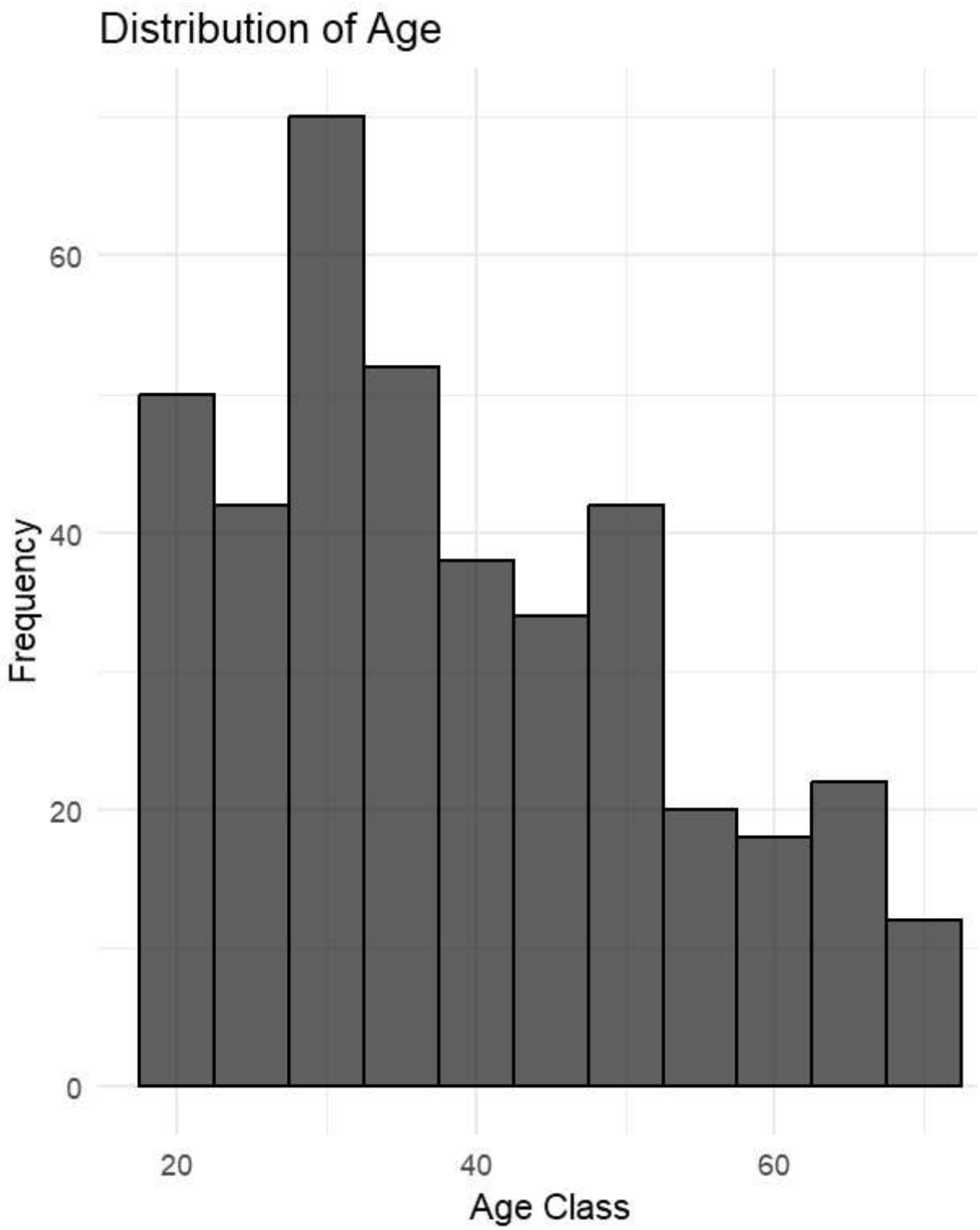

```
pie3D(a,labels=lbs,
    main="Pie Chart Depicting Ratio of Female and Male")
```



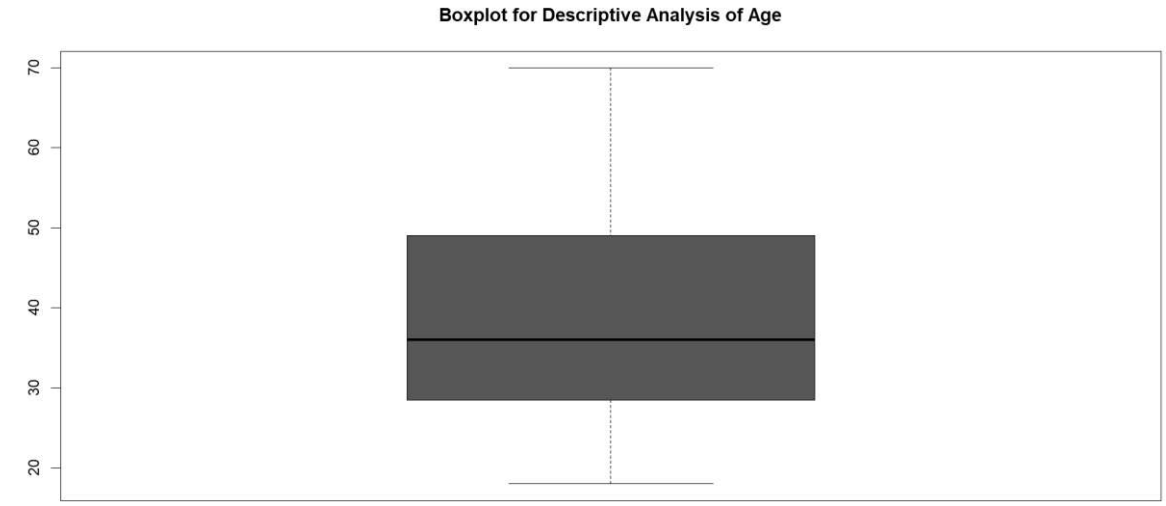
From the above graph, we conclude that the percentage of females is 56% whereas the percentage of male customers in the dataset is 44%.

Age distribution visualization:

```r
# Age Distribution
ggplot(customer_data, aes(x = Age)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Distribution of Age", x = "Age Class", y = "Frequency") +
  theme_minimal()
```

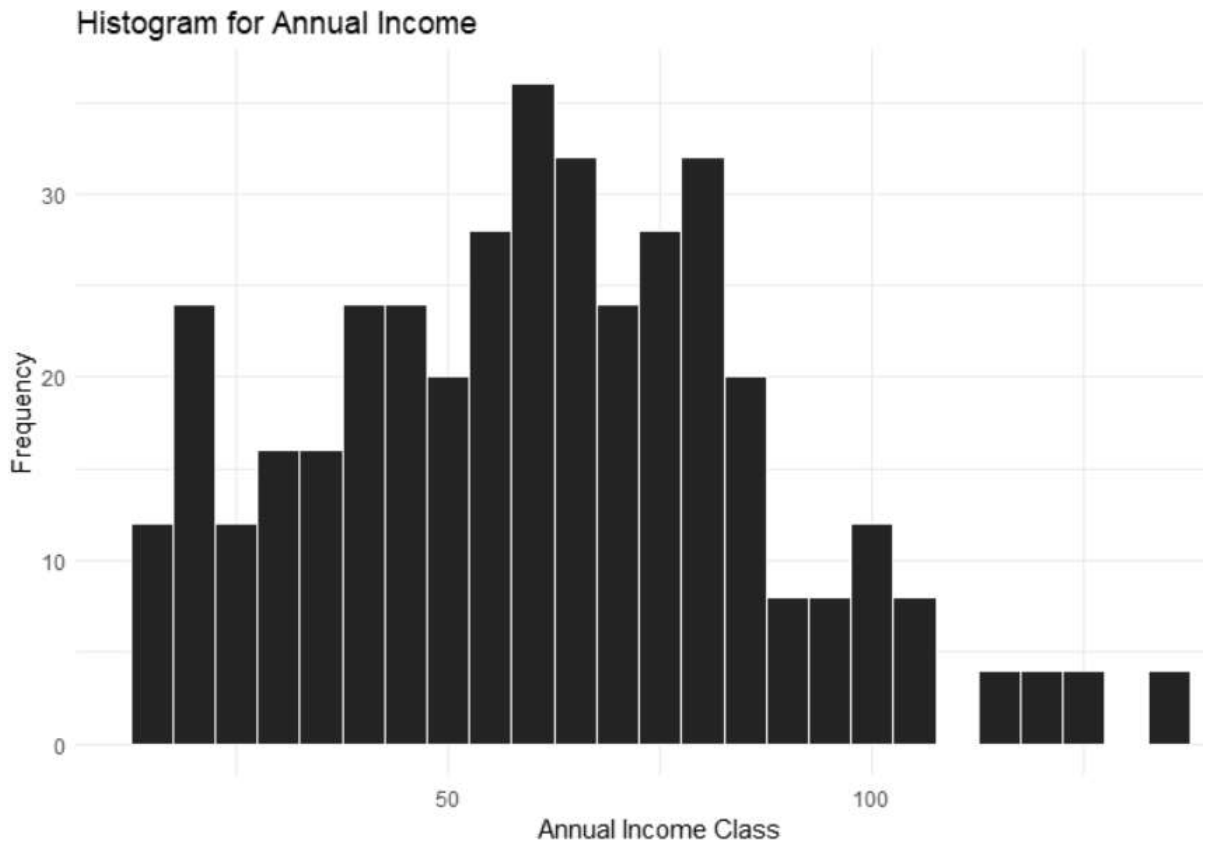


```
boxplot(customer_data$Age,
        col="#ff0066",
        main="Boxplot for Descriptive Analysis of Age")
```

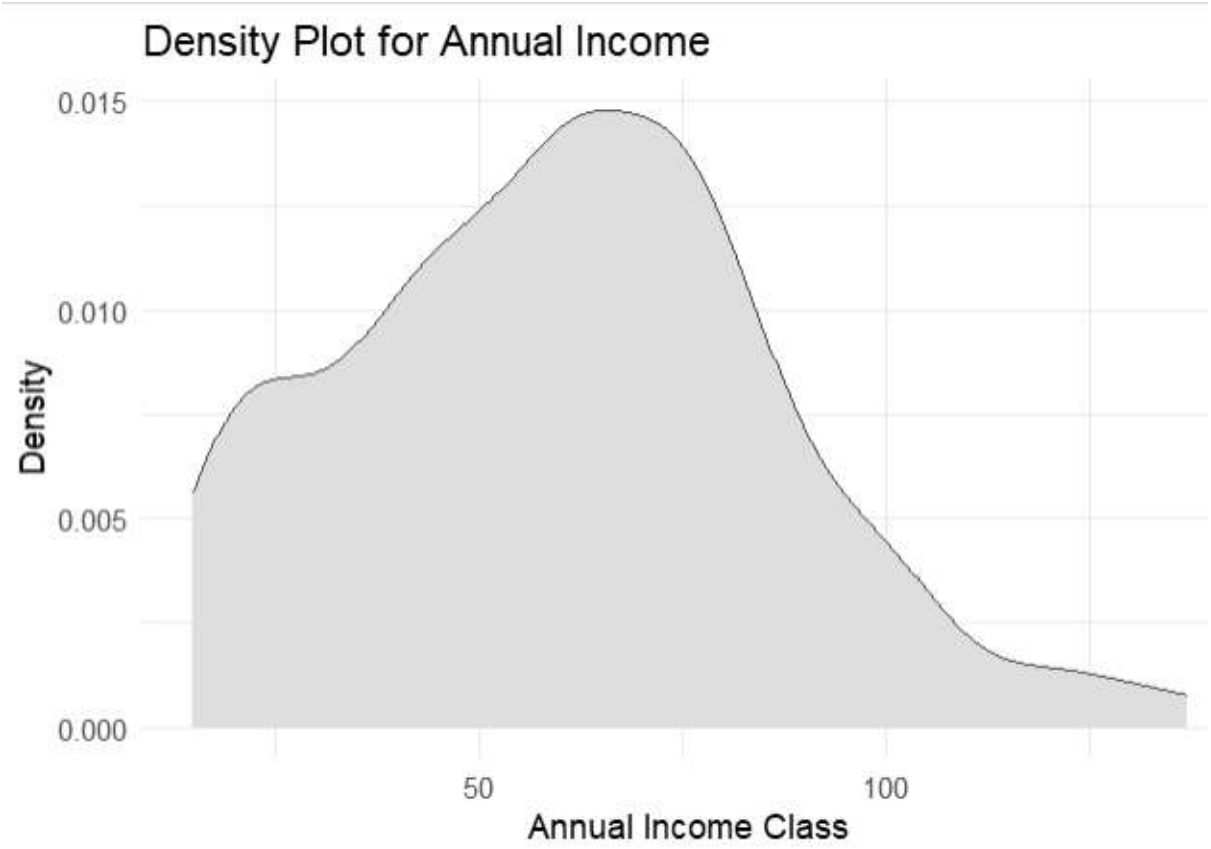


Annual Income

```
# Analysis of Annual Income
ggplot(customer_data, aes(x = Annual.Income..k..)) +
  geom_histogram(fill = "#660033", color = "white", binwidth = 5) +
  labs(title = "Histogram for Annual Income", x = "Annual Income Class", y = "Frequency") +
  theme_minimal()
```

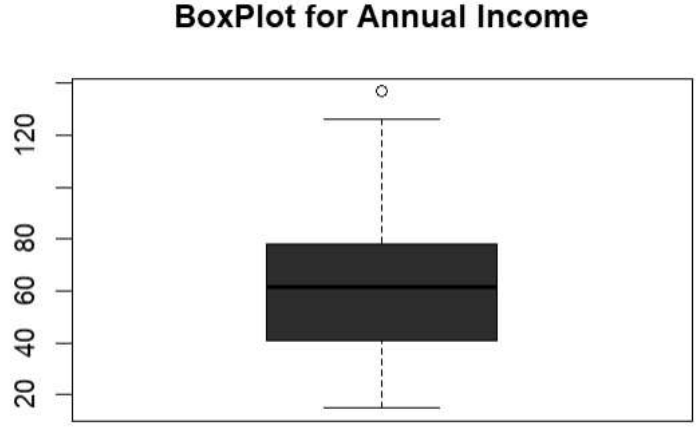


```r
ggplot(customer_data, aes(x = Annual.Income..k..)) +
  geom_density(fill = "#ccff66", color = "red") +
  labs(title = "Density Plot for Annual Income", x = "Annual Income Class", y = "Density") +
  theme_minimal()
```



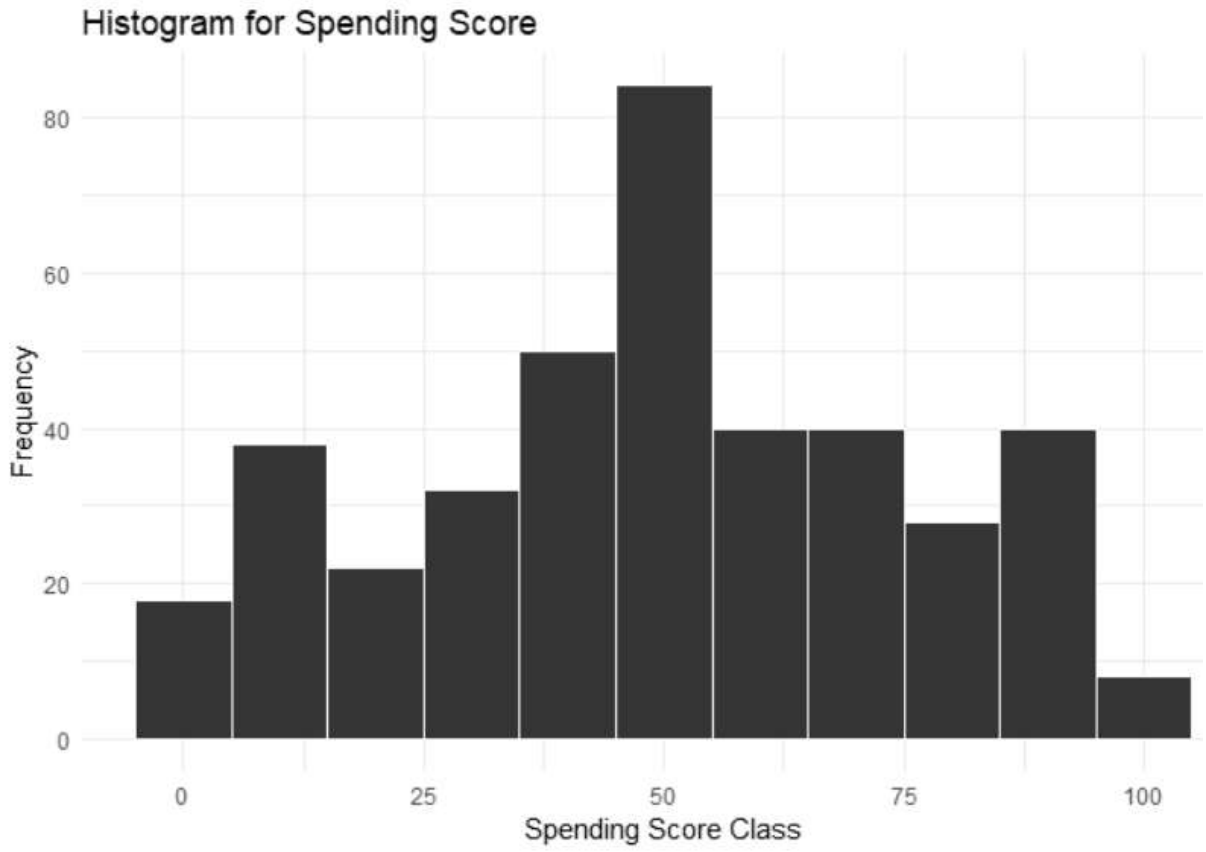

```r
boxplot(customer_data$Annual.Income..k..,

        col="#990000",
        main="BoxPlot for Annual Income")
```



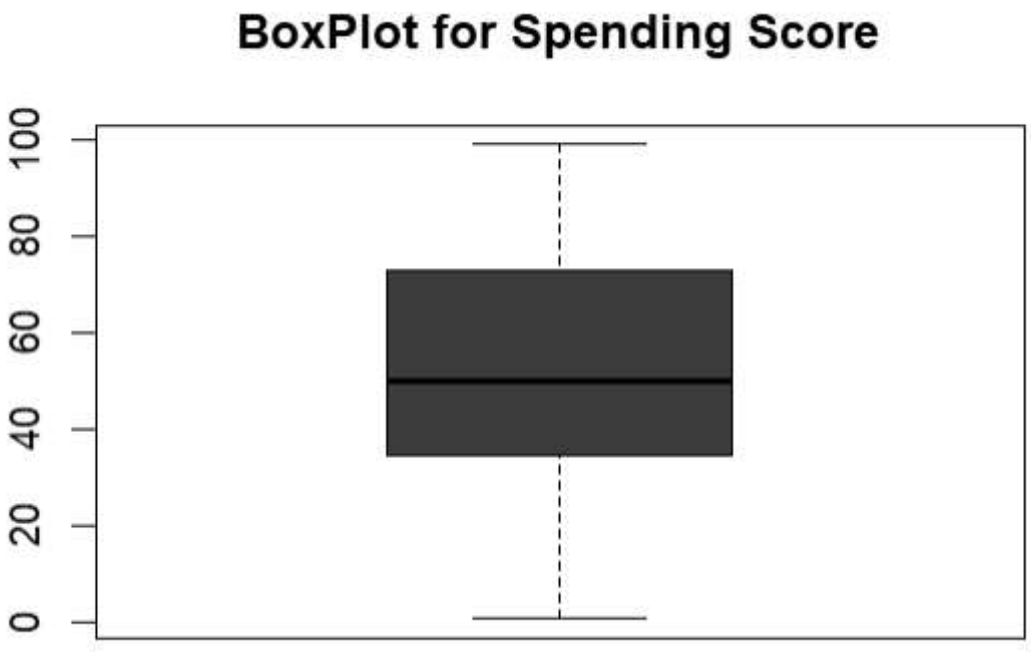
Spending Score

```
# Spending Score Analysis
ggplot(customer_data, aes(x = Spending.Score..1.100.)) +
  geom_histogram(fill = "#6600cc", color = "white", binwidth = 10) +
  labs(title = "Histogram for Spending Score", x = "Spending Score Class", y = "Frequency") +
  theme_minimal()
```




```
boxplot(customer_data$Spending.Score..1.100.,

        col="#990000",
        main="BoxPlot for Descriptive Analysis of Spending Score")
```




```
 99  library(purrr)
100  set.seed(123)
101  # function to calculate total intra-cluster sum of square
102  iss <- function(k) {
103    kmeans(customer_data[,3:5],k,iter.max=100,nstart=100,algorithm="Lloyd" )$tot.withinss
104  }
105
106  k.values <- 1:10
107
108
109  iss_values <- map_dbl(k.values, iss)
110
111  plot(k.values, iss_values,
112       type="b", pch = 19, frame = FALSE,
113       xlab="Number of clusters K",
114       ylab="Total intra-clusters sum of squares")
```

K-means Algorithm:

The K-means algorithm is a popular unsupervised machine learning technique used for clustering data points into groups, or clusters, based on similarity. Here's a breakdown of how it works:

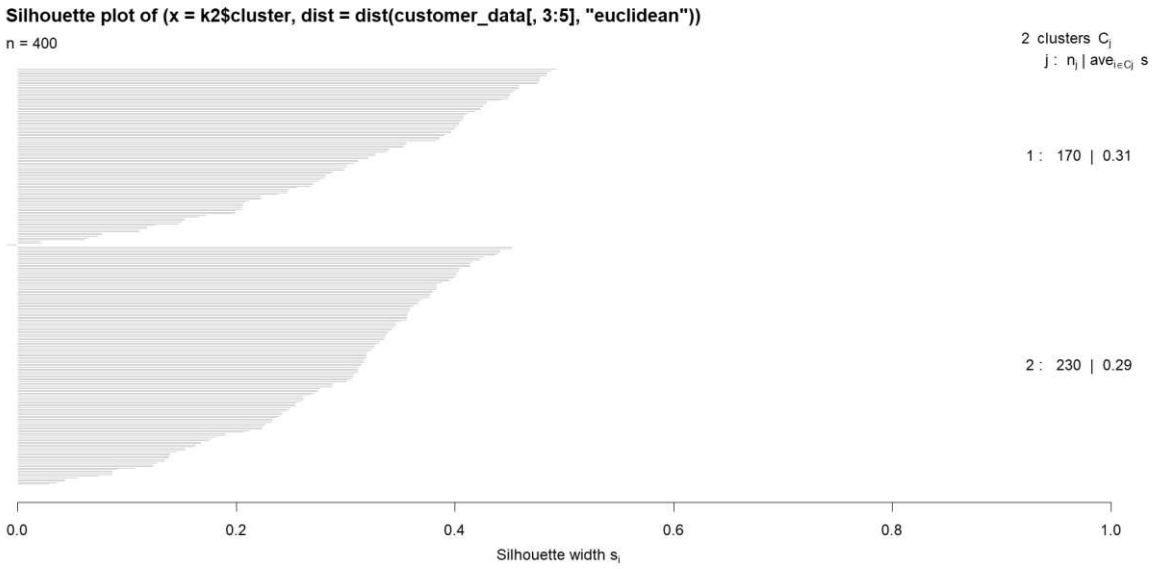
1. Initialization: The algorithm begins by randomly selecting K data points from the dataset to serve as the

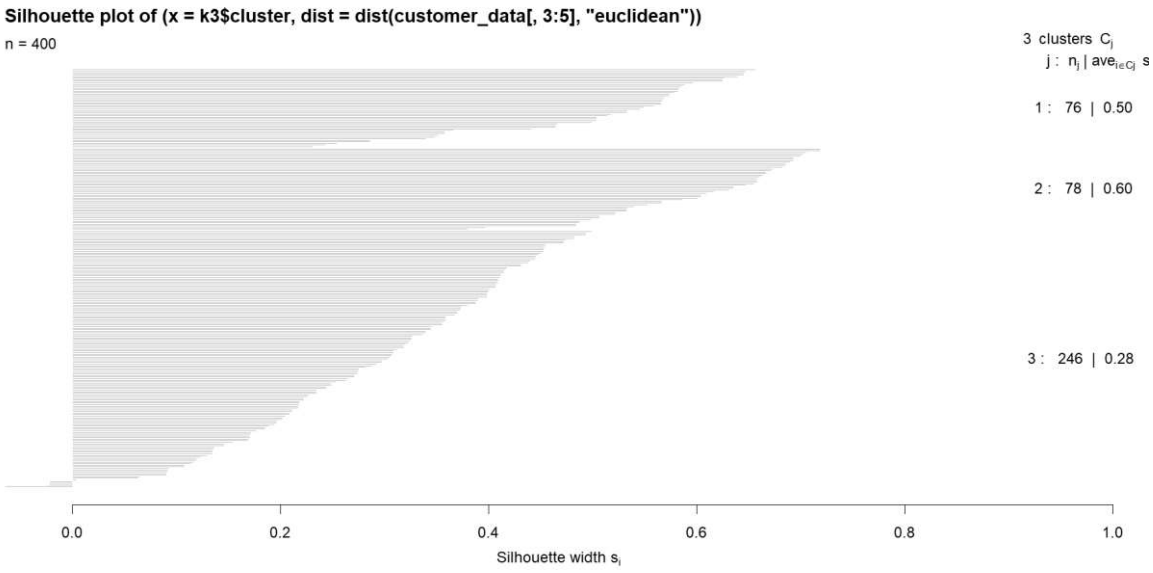
initial centroids of the clusters. These centroids can be randomly chosen or based on some predefined criteria.

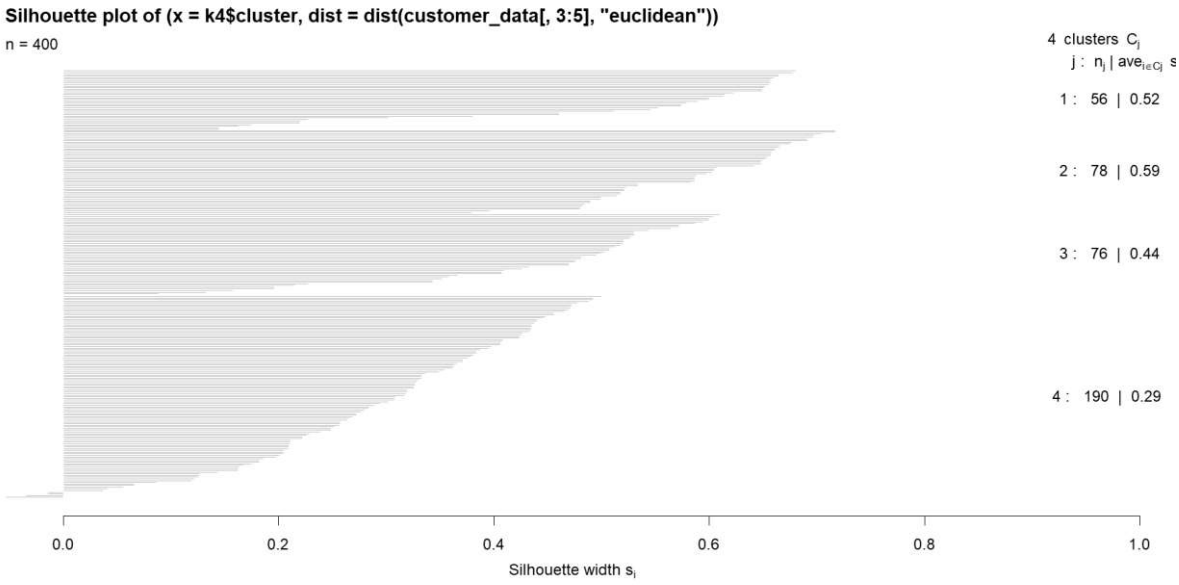
2. Assignment: Each data point is then assigned to the nearest centroid based on a distance metric, typically Euclidean distance. This step forms K clusters.
3. Update: After assigning each data point to a cluster, the centroids are recomputed based on the mean of all data points assigned to each cluster. This new centroid becomes the center of its respective cluster.
4. Repeat: Steps 2 and 3 are repeated iteratively until convergence, meaning that the centroids no longer change significantly, or a predefined number of iterations is reached.
5. Convergence: The algorithm converges when the centroids stabilize and no longer change significantly between iterations.

   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.

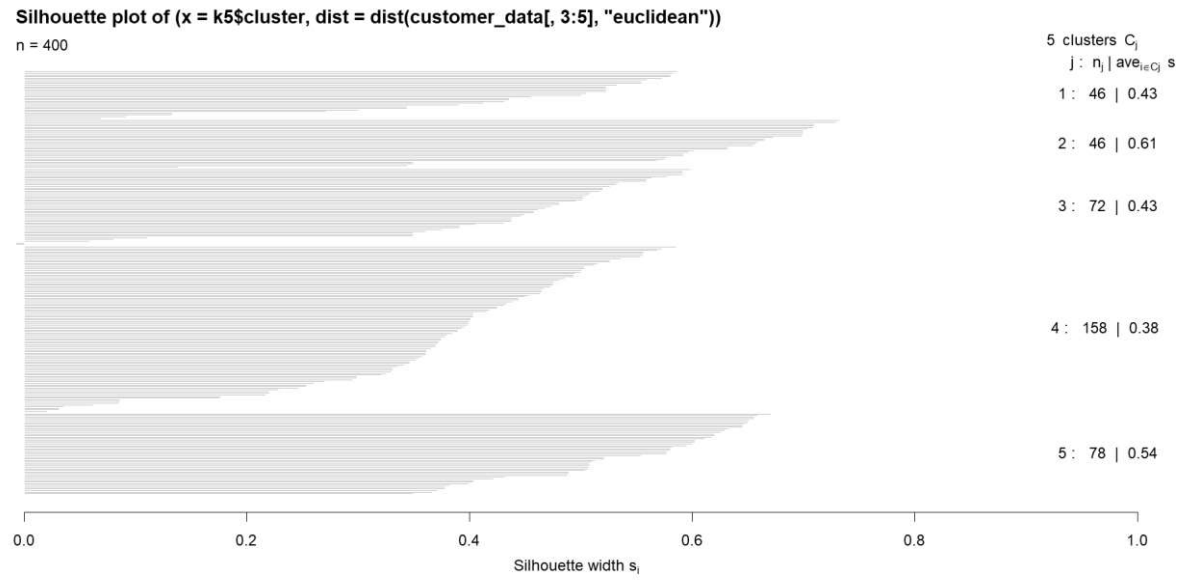
#Average Silhouette Method:


Silhouette plot of (x = k2$cluster, dist = dist(customer_data[, 3:5], "euclidean"))
n = 400
2 clusters C_j
j : n_j | $ave_{i \in Cj}$ s
1 : 170 | 0.31
2 : 230 | 0.29
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.3

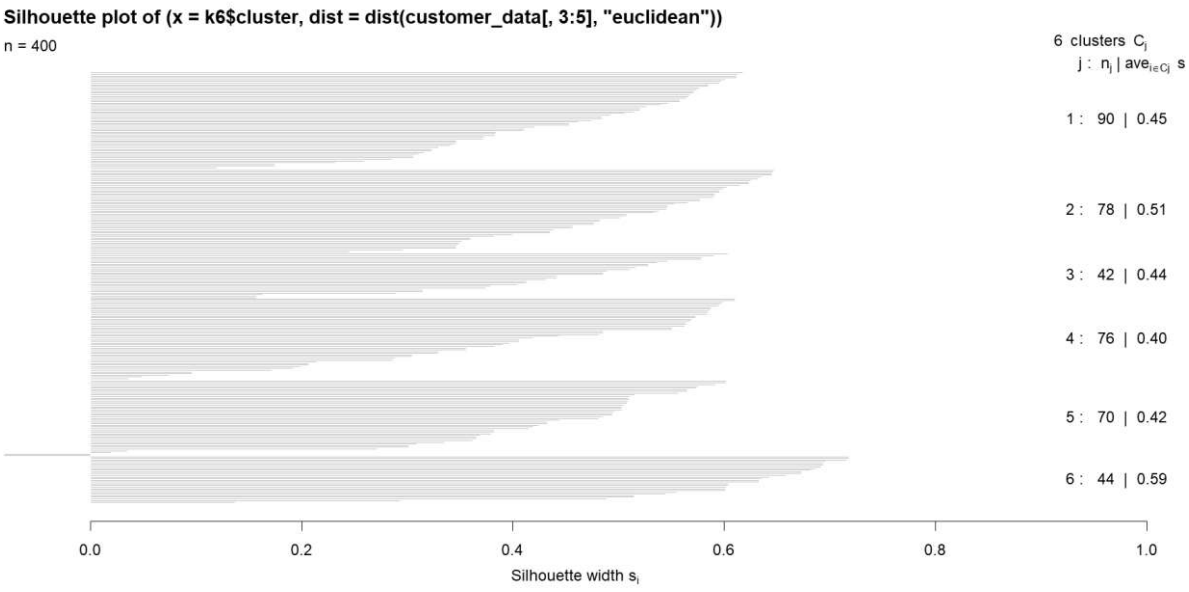

Silhouette plot of (x = k3$cluster, dist = dist(customer_data[, 3:5], "euclidean"))
n = 400
3 clusters C_j
j : n_j | $ave_{i \in Cj}$ s
1 : 76 | 0.50
2 : 78 | 0.60
3 : 246 | 0.28
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.39

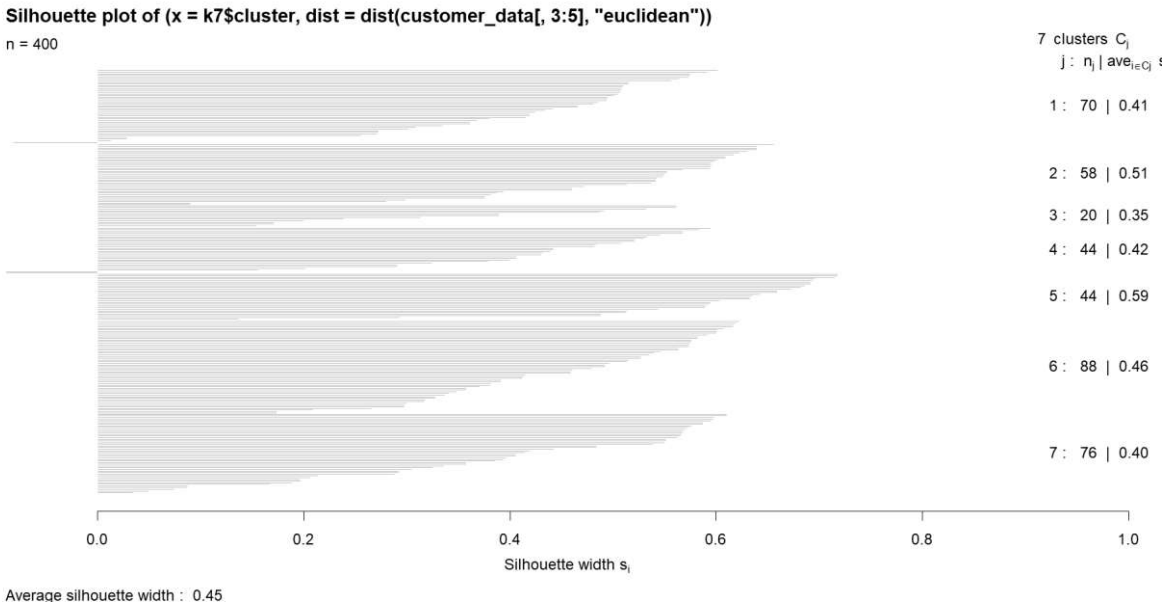


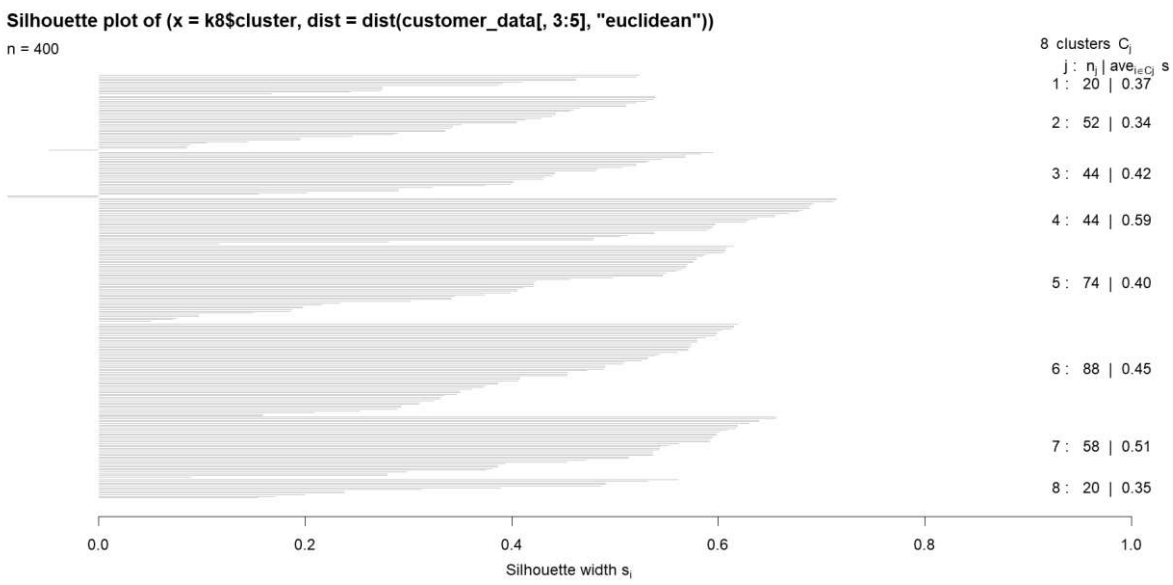
Average silhouette width : 0.41



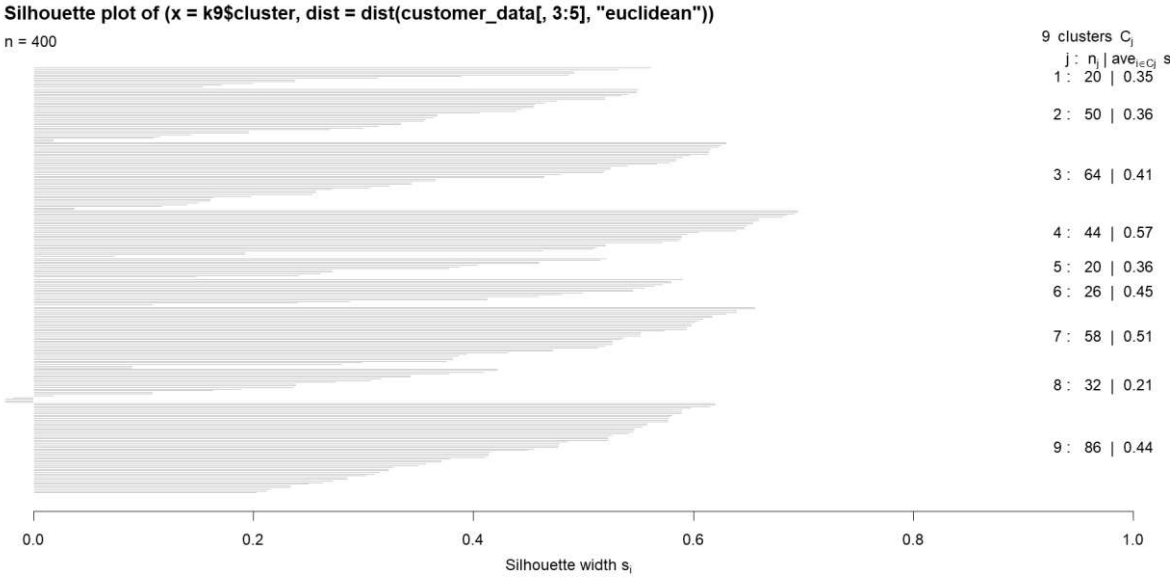
Average silhouette width : 0.45


Silhouette plot of (x = k6$cluster, dist = dist(customer_data[, 3:5], "euclidean"))
n = 400
6 clusters C_j
j : n_j | $ave_{i \in Cj}$ s
1 : 90 | 0.45
2 : 78 | 0.51
3 : 42 | 0.44
4 : 76 | 0.40
5 : 70 | 0.42
6 : 44 | 0.59
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.46

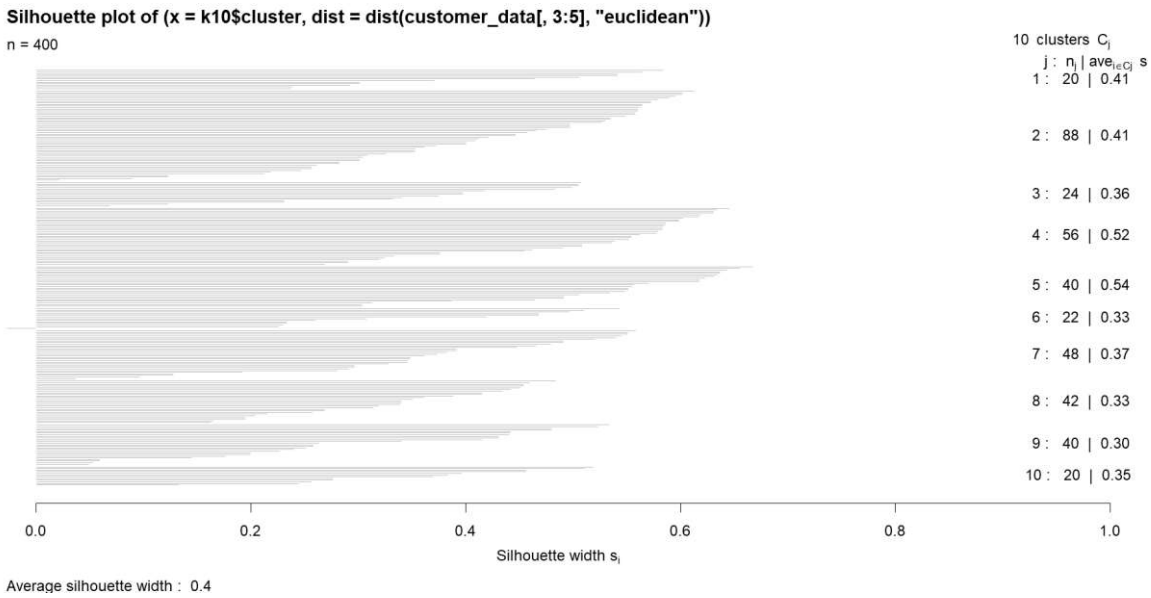

Silhouette plot of (x = k7$cluster, dist = dist(customer_data[, 3:5], "euclidean"))
n = 400
7 clusters C_j
j : n_j | $ave_{i \in Cj}$ s
1 : 70 | 0.41
2 : 58 | 0.51
3 : 20 | 0.35
4 : 44 | 0.42
5 : 44 | 0.59
6 : 88 | 0.46
7 : 76 | 0.40
0.0
0.2
0.4
0.6
0.8
1.0
Silhouette width s_i
Average silhouette width : 0.45

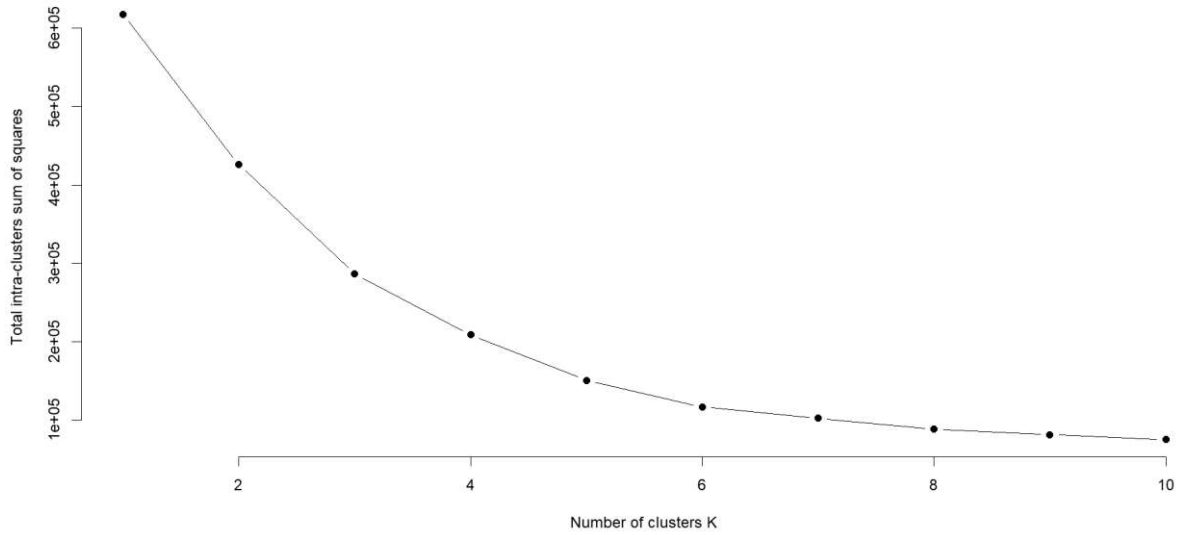


Average silhouette width : 0.44

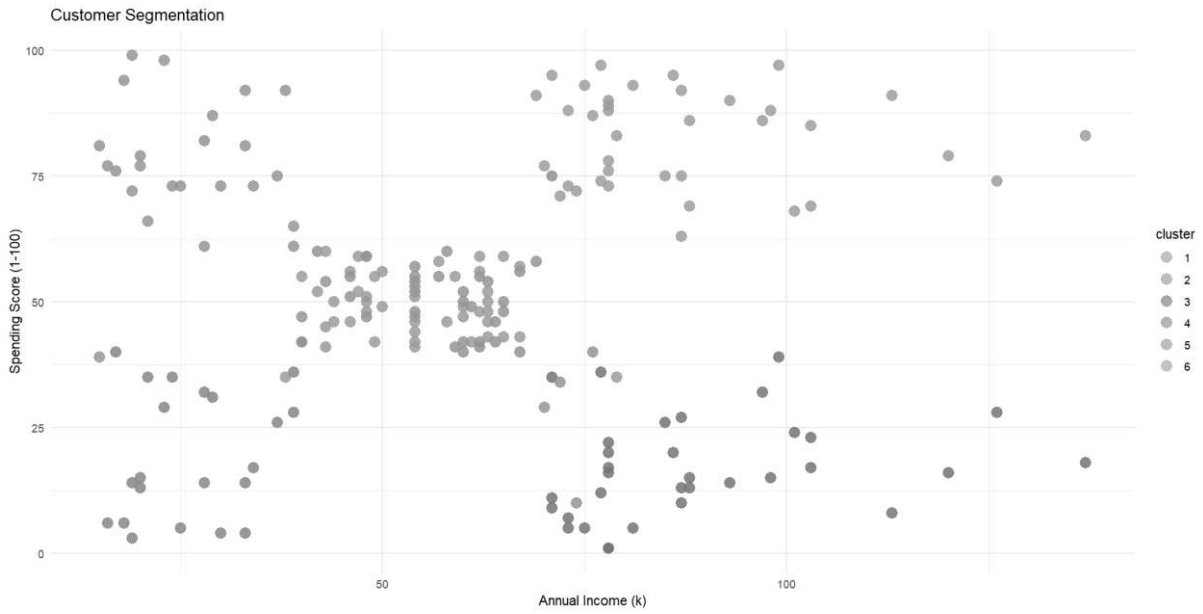


Average silhouette width : 0.42





Plotting the K means Clustering:



Cluster 1 and 2– These clusters represent the customer_data with the medium income salary as well as the medium annual spend of salary.

Cluster 6 – 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 4 – 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.

Performing hierarchal clustering:

Hierarchical clustering is another popular method in unsupervised machine learning used to group similar data points into clusters. Unlike K-means, hierarchical clustering does not require the number of clusters to be predefined. Instead, it creates a hierarchical tree of clusters known as a dendrogram.

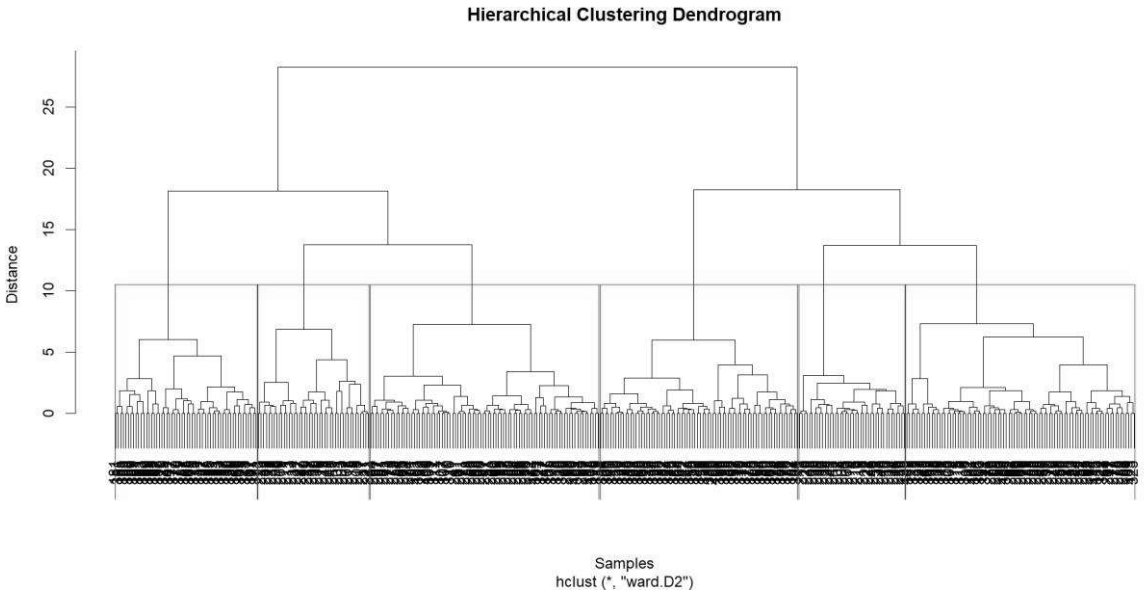
Hierarchical clustering algorithm:

1. Initialization: Each data point starts as its own cluster, so there are as many clusters as there are data points.
2. Distance Calculation: A distance matrix is computed, representing the similarity or dissimilarity between each pair of data points. Common distance metrics include Euclidean distance, Manhattan distance, or correlation distance.
3. Merge or Split: The algorithm iteratively merges or splits clusters based on their similarity or dissimilarity until a termination criterion is met.

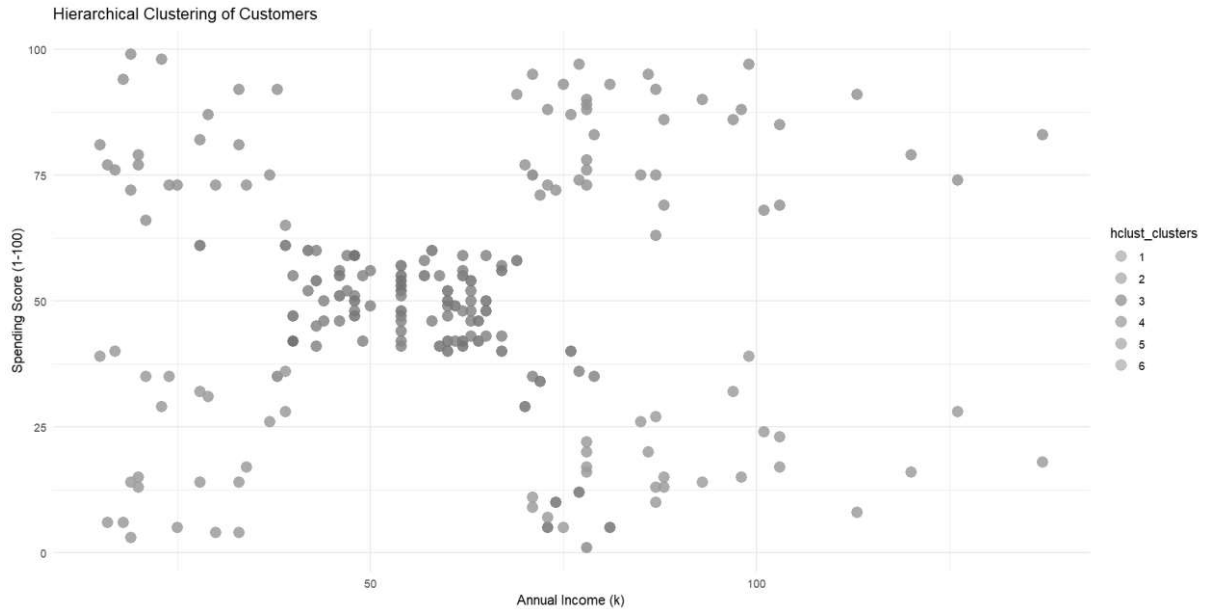
- Agglomerative Hierarchical Clustering: This is a bottom-up approach where each data point starts in its own cluster, and pairs of clusters are merged based on a similarity measure until all clusters are merged into one. The result is a dendrogram that shows the hierarchical relationships between clusters.
- Divisive Hierarchical Clustering: This is a top-down approach where all data points start in one cluster, and the algorithm recursively splits the cluster into smaller clusters until each data point is in its own cluster. However, divisive clustering is less common in practice.

4. Dendrogram Construction: As clusters are merged or split, a dendrogram is constructed to visualize the hierarchical relationships between clusters. The height at which two clusters are merged in the dendrogram represents their dissimilarity.
5. Termination: The termination criterion can be based on reaching a specific number of clusters, a threshold distance, or another predefined criterion.

Dendogram:



Hierarchichal clustering :



Conclusion:

From the customer segmentation analysis performed in the code, you can learn the following:

1. Insights into different customer segments:
    - The 6 clusters represent distinct groups of customers based on their annual income and spending score.
    - Each cluster has unique characteristics, allowing you to understand the different types of customers the business is serving.
2. Customer behavior patterns:
    - The clustering reveals how customers' annual income and spending behaviors are distributed and correlated.

- You can identify patterns, such as high-income customers with low spending, or low-income customers with high spending.

3. Potential target markets:
    - The clusters can help you identify the most valuable or profitable customer segments for the business.
    - You can focus on the clusters with high annual income and high spending scores, as these customers are likely the most valuable.
4. Tailored marketing strategies:
    - Understanding the distinct customer segments allows you to develop targeted marketing campaigns, product offerings, and customer engagement strategies for each group.
    - By catering to the specific needs and preferences of each cluster, you can improve customer satisfaction and loyalty.

The main conclusion you can draw from this customer segmentation analysis is that the business has a diverse customer base with varying levels of annual income and spending behavior. By identifying these distinct clusters, the business can better understand its customer base, prioritize its resources, and develop more effective strategies to serve each segment's needs.

RESULT:

This information can be used to make more informed decisions regarding product development, pricing, marketing, and customer relationship management. Overall, the customer segmentation analysis provides valuable insights that can help the business optimize its operations and enhance its competitiveness in the market.

REFERENCE:

- https://www.analyticsvidhya.com/blog/2021/05/k-mean-getting-the-optimal-number-of-clusters/
- www.wikipedia.com
- https://www.geeksforgeeks.org/k-means-clustering-introduction/
- https://www.javatpoint.com/hierarchical-clustering-in-machine-learning

APPENDIX:

R CODE:

```
# Load libraries
library(ggplot2)
library(cluster)
library(gridExtra)
library(grid)
library(NbClust)
library(factoextra)
library(plotrix)
library(plotly)


# Load data
customer_data=read.csv("C:\\Users\\vedan\\OneDrive\\Desktop\\Rda2\\Mall_Customers.csv")


# Check data structure and summary stats
str(customer_data)
head(customer_data)
summary(customer_data$Age)
sd(customer_data$Age)
summary(customer_data$Annual.Income..k..)
sd(customer_data$Annual.Income..k..)
summary(customer_data$Spending.Score..1.100.)
sd(customer_data$Spending.Score..1.100.)
print(paste("Number of null values:", sum(is.na(customer_data))))
```

```
# Visualization of Gender Distribution
a <- table(customer_data$Gender)
g<-ggplot(customer_data, aes(x = Gender)) +
  geom_bar(aes(fill = Gender), width = 0.5) +
  geom_text(stat='count', aes(label=..count.., y=..count..), vjust=-0.5) +
  scale_fill_manual(values = rainbow(2)) +
  labs(title = "Counts of Male vs Female", x = "Gender", y = "Count") +
  theme_minimal()
ggplotly(g)
library(plotrix)
pie3D(a,labels=lbs,
      main="Pie Chart Depicting Ratio of Female and Male")

# Age Distribution
g2<-ggplot(customer_data, aes(x = Age)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black", alpha = 0.7) +
  labs(title = "Distribution of Age", x = "Age Class", y = "Frequency") +
  theme_minimal()
ggplotly(g2)
boxplot(customer_data$Age,
        col="#ff0066",
        main="Boxplot for Descriptive Analysis of Age")


# Analysis of Annual Income
g3<-ggplot(customer_data, aes(x = Annual.Income..k..)) +
  geom_histogram(fill = "#660033", color = "white", binwidth = 5) +
```

```
  labs(title = "Histogram for Annual Income", x = "Annual Income Class", y = "Frequency") +
  theme_minimal()
ggplotly(g3)

g4<-ggplot(customer_data, aes(x = Annual.Income..k..)) +
  geom_density(fill = "#ccff66", color = "red") +
  labs(title = "Density Plot for Annual Income", x = "Annual Income Class", y = "Density") +
  theme_minimal()
ggplotly(g4)
boxplot(customer_data$Annual.Income..k..,

        col="#990000",
        main="BoxPlot for Annual Income")



# Spending Score Analysis
g4<-ggplot(customer_data, aes(x = Spending.Score..1.100.)) +
  geom_histogram(fill = "#6600cc", color = "white", binwidth = 10) +
  labs(title = "Histogram for Spending Score", x = "Spending Score Class", y = "Frequency") +
  theme_minimal()
ggplotly(g4)
boxplot(customer_data$Spending.Score..1.100.,

        col="#990000",
        main="BoxPlot for Descriptive Analysis of Spending Score")
```

```
#Finding optimal number of clusters:
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")



#Average Silhouette Method



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")))
```

```
#K-means Algorithm



# K-means Clustering
data_for_clustering_scaled <- scale(customer_data[,3:5])
max_clusters <- 10
sse <- numeric(max_clusters)
for (k in 1:max_clusters) {
  set.seed(123)
  model <- kmeans(customer_data[,3:5], centers = k, nstart = 25)
  sse[k] <- model$tot.withinss
}

# Attaching cluster results to the original data for visualization
customer_data$cluster <- as.factor(kmeans_result$cluster)


# Plotting clusters
g4<-ggplot(customer_data, aes(x = Annual.Income..k.., y =
Spending.Score..1.100., color = cluster)) +
  geom_point(alpha = 0.6, size = 4) +
  labs(title = "Customer Segmentation", x = "Annual Income (k)", y =
"Spending Score (1-100)") +
  theme_minimal()
ggplotly(g4)
# Calculate distance matrix
d <- dist(data_for_clustering_scaled, method = "euclidean")
```

```
# Perform hierarchical clustering
hc <- hclust(d, method = "ward.D2")

# Plot the dendrogram
plot(hc, main = "Hierarchical Clustering Dendrogram", xlab = "Samples", ylab = "Distance")
rect.hclust(hc, k = 6, border = "red")  # Assuming you choose 5 clusters

# Cut tree to create 6 clusters
clusters <- cutree(hc, k = 6)
customer_data$hclust_clusters <- as.factor(clusters)

# Plot clusters
g6<-ggplot(customer_data, aes(x = Annual.Income..k.., y = Spending.Score..1.100., color = hclust_clusters)) +
  geom_point(alpha = 0.6, size = 4) +
  labs(title = "Hierarchical Clustering of Customers", x = "Annual Income (k)", y = "Spending Score (1-100)") +
  theme_minimal()
ggplotly(g6)
```

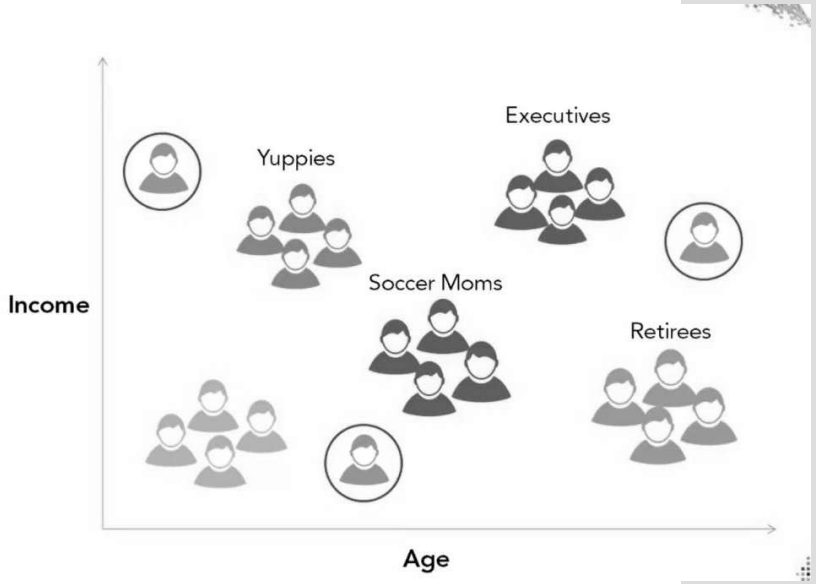
Customer segmentation

- By
- Heet Shah (21BCE2938)
- Vedanta Sharan (21BCE2970)

Introduction

WHAT IS CUSTOMER SEGMENTATION?

- Process of dividing a company's customers into groups based on common characteristics
- A company might segment customers based on a wide range of factors, including:
  - Products previously purchased from the company
  - Location
  - Age
  - Gender



OBJECTIVE

- Segment customers into different groups using various clustering algorithms

Data Preperation

Dataset

	A	B	C	D	E
1	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
2	1	Male	19	15	39
3	2	Male	21	15	81
4	3	Female	20	16	6
5	4	Female	23	16	77
6	5	Female	31	17	40
7	6	Female	22	17	76
8	7	Female	35	18	6
9	8	Female	23	18	94
10	9	Male	64	19	3
11	10	Female	30	19	72
12	11	Male	67	19	14
13	12	Female	35	19	99
14	13	Female	58	20	15
15	14	Female	24	20	77
16	15	Male	37	20	13
17	16	Male	22	20	79

Summary of Dataset

- Structure of Data

```
> str(customer_data)
'data.frame':   400 obs. of  5 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 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 ...
```

- Head of Data

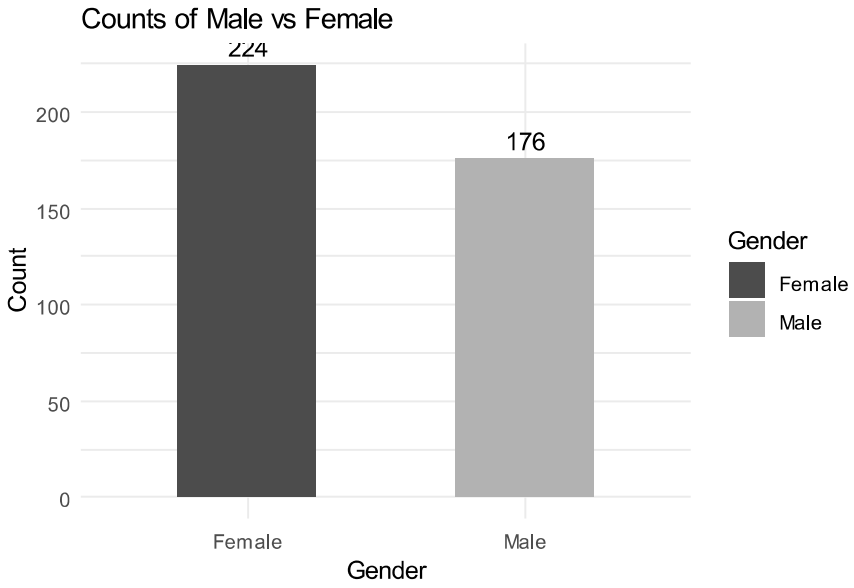
```
> head(customer_data)
  CustomerID Gender Age Annual.Income..k.. Spending.Score..1.100.
1          1   Male  19                 15                      39
2          2   Male  21                 15                      81
3          3 Female  20                 16                       6
4          4 Female  23                 16                      77
5          5 Female  31                 17                      40
6          6 Female  22                 17                      76
```

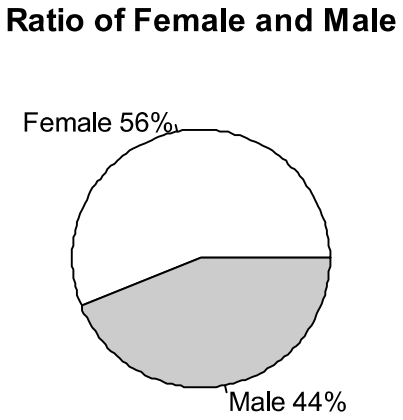
- Column wise Summary of Data

```
> summary(customer_data)
   CustomerID       Gender                Age        Annual.Income..k.. Spending.Score..1.100.
 Min.   :  1.0   Length:400         Min.   :18.00   Min.   : 15.00     Min.   : 1.00
 1st Qu.:100.8   Class :character   1st Qu.:28.75   1st Qu.: 41.50     1st Qu.:34.75
 Median :200.5   Mode  :character   Median :36.00   Median : 61.50     Median :50.00
 Mean   :200.5                      Mean   :38.85   Mean   : 60.56     Mean   :50.20
 3rd Qu.:300.2                      3rd Qu.:49.00   3rd Qu.: 78.00     3rd Qu.:73.00
 Max.   :400.0                      Max.   :70.00   Max.   :137.00     Max.   :99.00
```

Visualization of Data

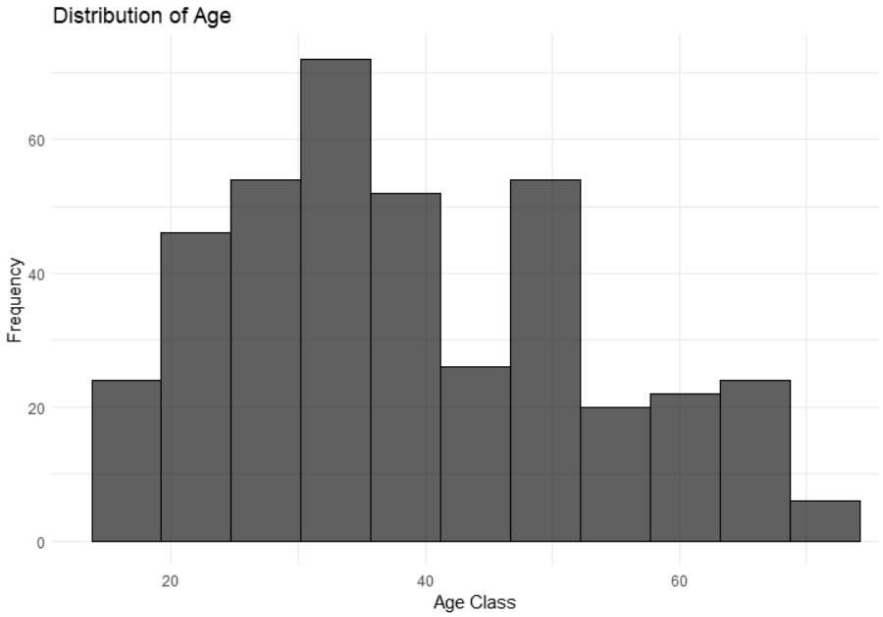
- Distribution of Gender



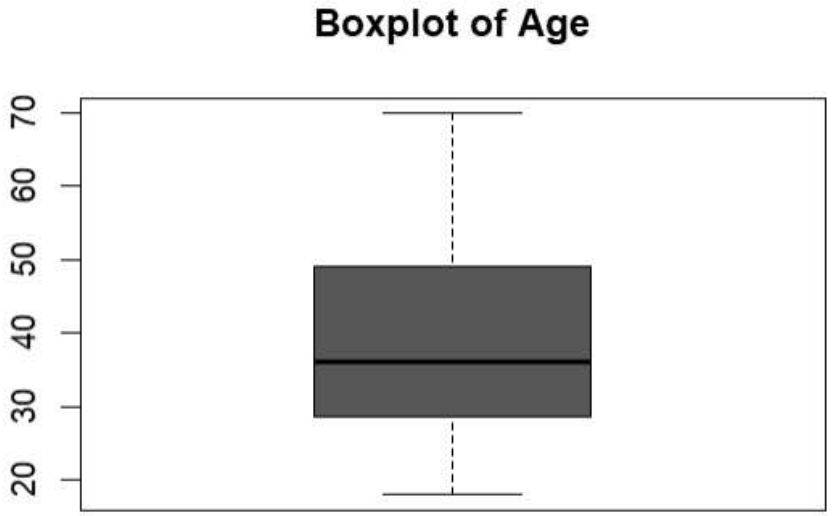


Visualization of Data

- Distribution of Age

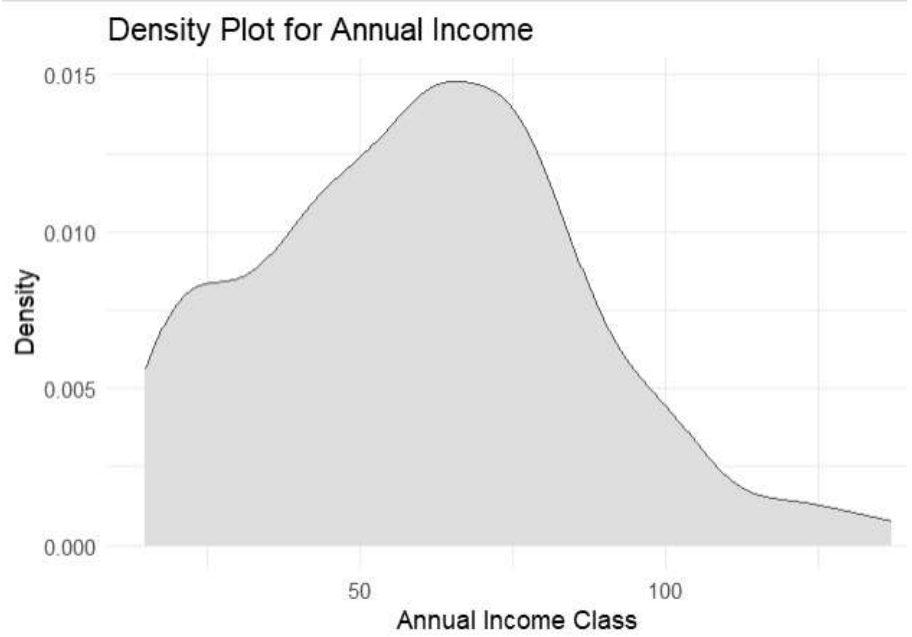


- Finding outliers using box plot

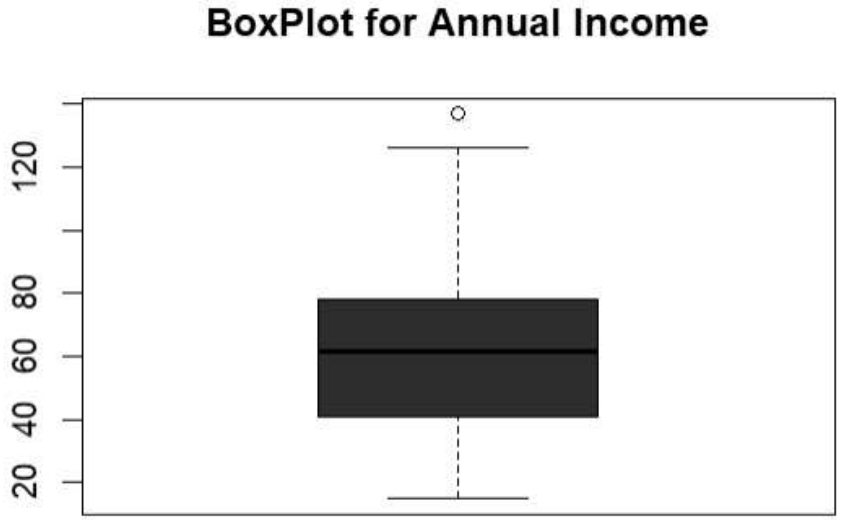


Visualization of Data

- Distribution of Annual Income

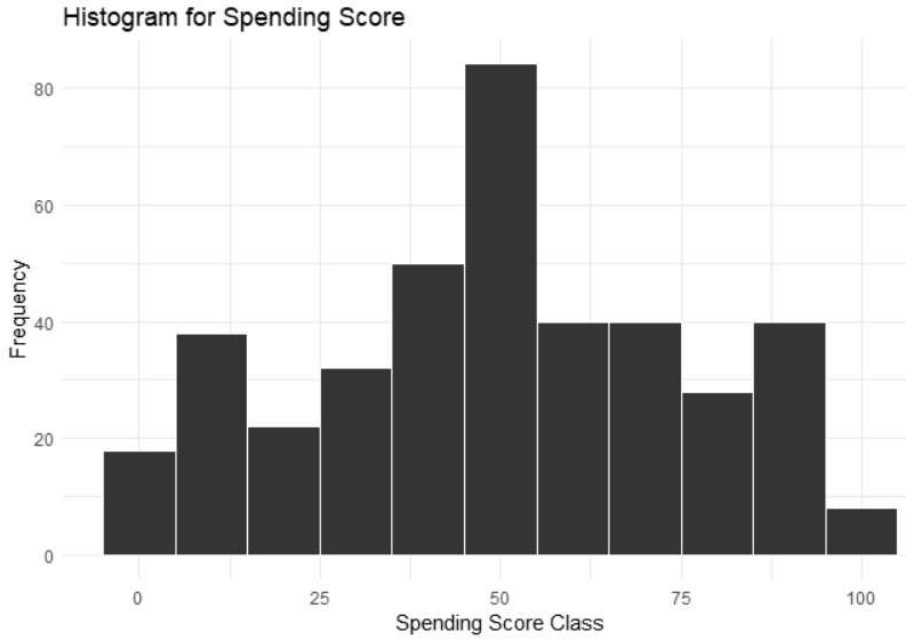


- Finding outliers using box plot

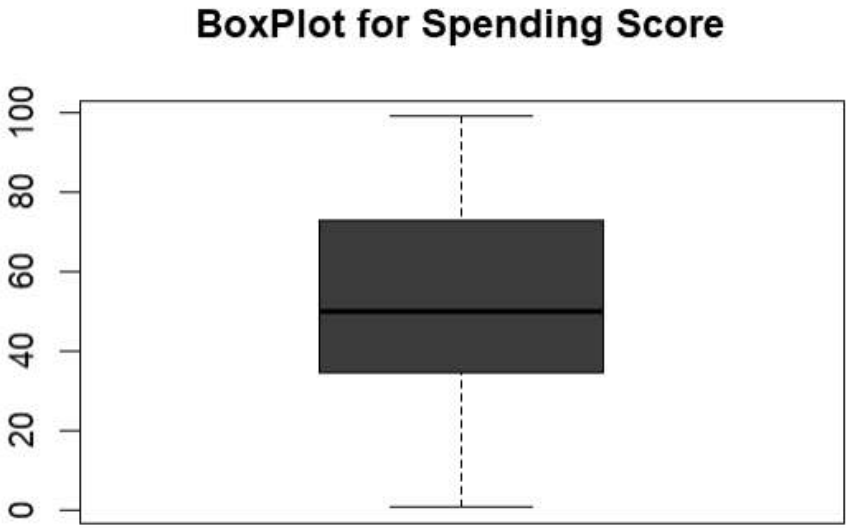


Visualization of Data

- Distribution of Spending Score

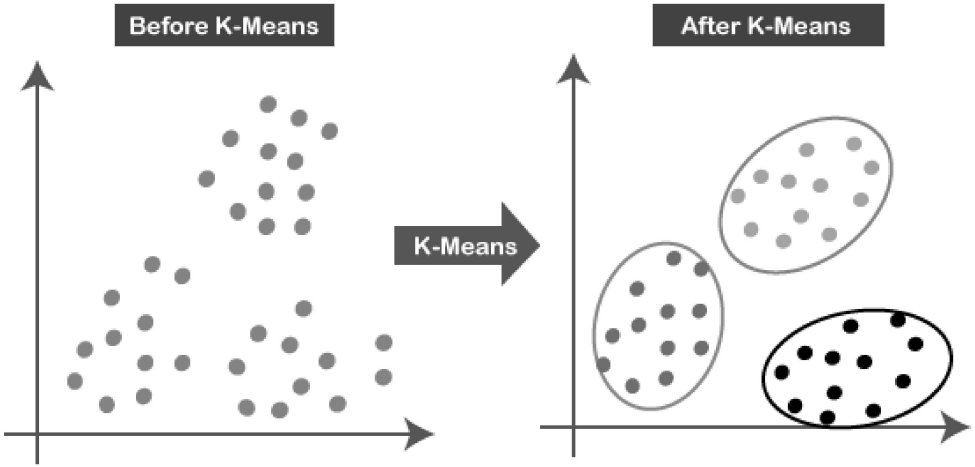


- Finding outliers using box plot

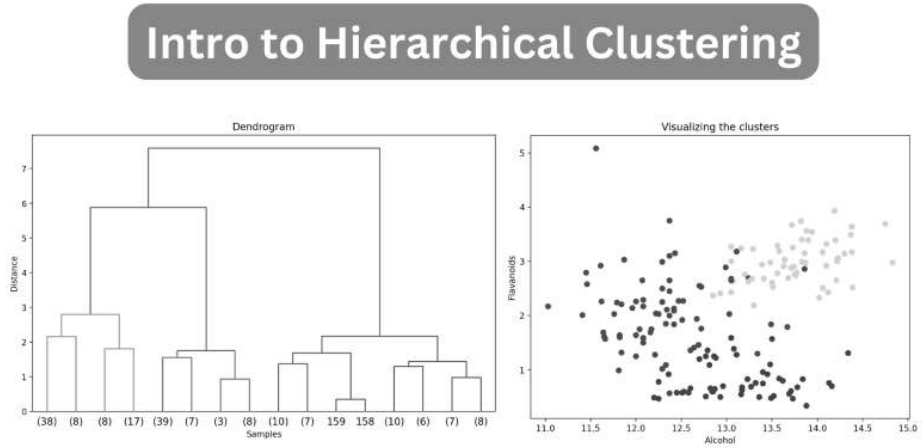


Clustering Algorithms

- K-Means Clustering
  - Divides data into a predefined number of clusters by assigning each point to the nearest cluster center
  - optimizing the positions of the centers to minimize the within-cluster variance

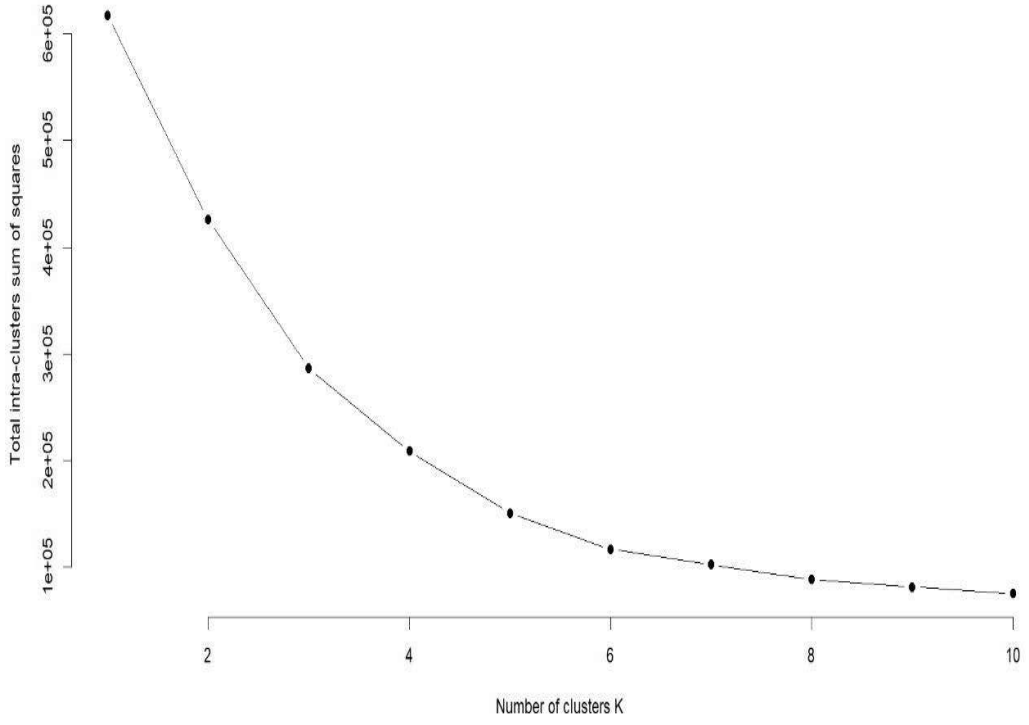


- Hierarchical Clustering
  - Builds a hierarchy of clusters either by successively merging smaller clusters into larger ones (agglomerative approach)
  - Visualized using a dendrogram



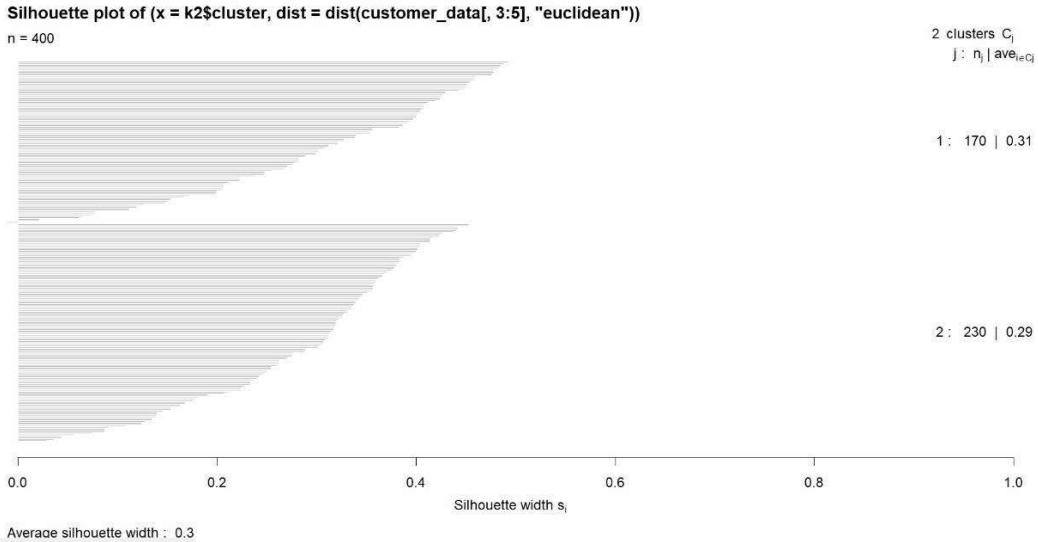
Determining Optimal Number of Clusters

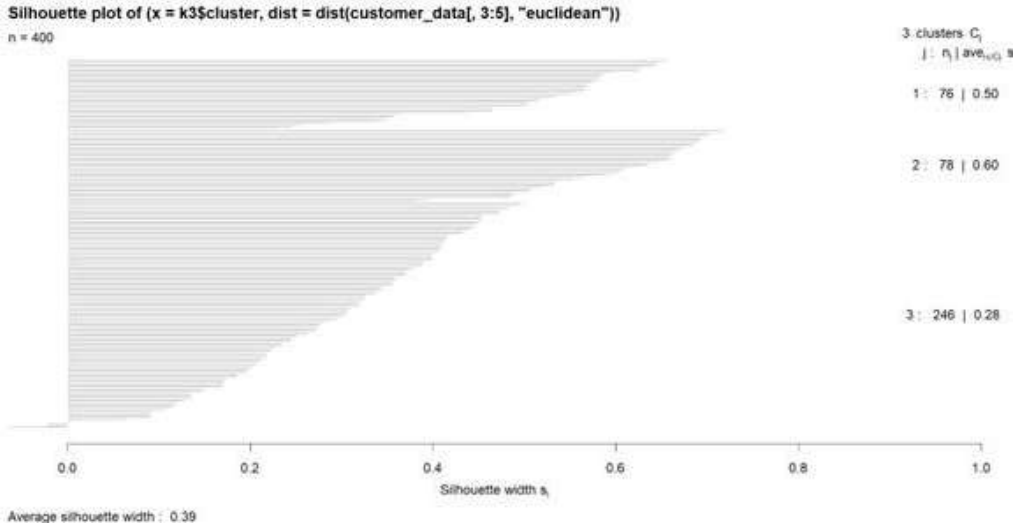
- Elbow Method
  - Calculate the Sum of Squares of all the points w.r.t the cluster center and calculate the average for all the distance.
  - The optimal cluster is where we get bend in the plot.



Determining Optimal Number of Clusters

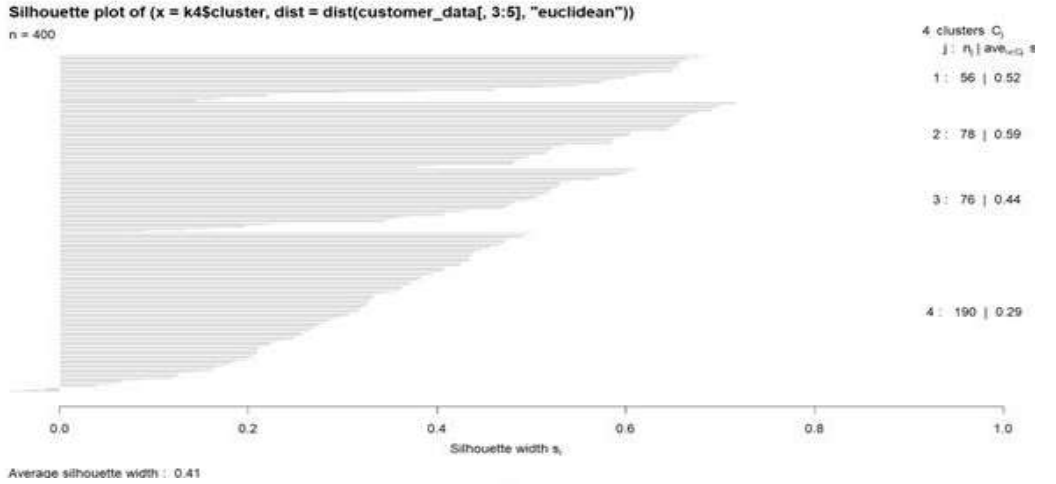
- Silhoutte Plots
  - Calculate the Sum of Squares of all the points w.r.t the cluster center and plot those values.
  - The optimal cluster which one has the highest silhouette width.

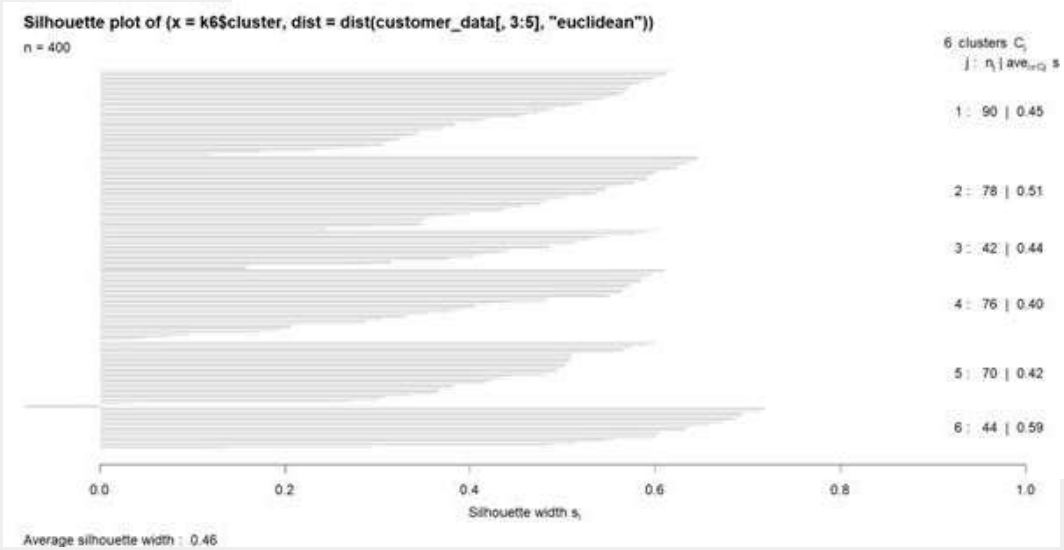


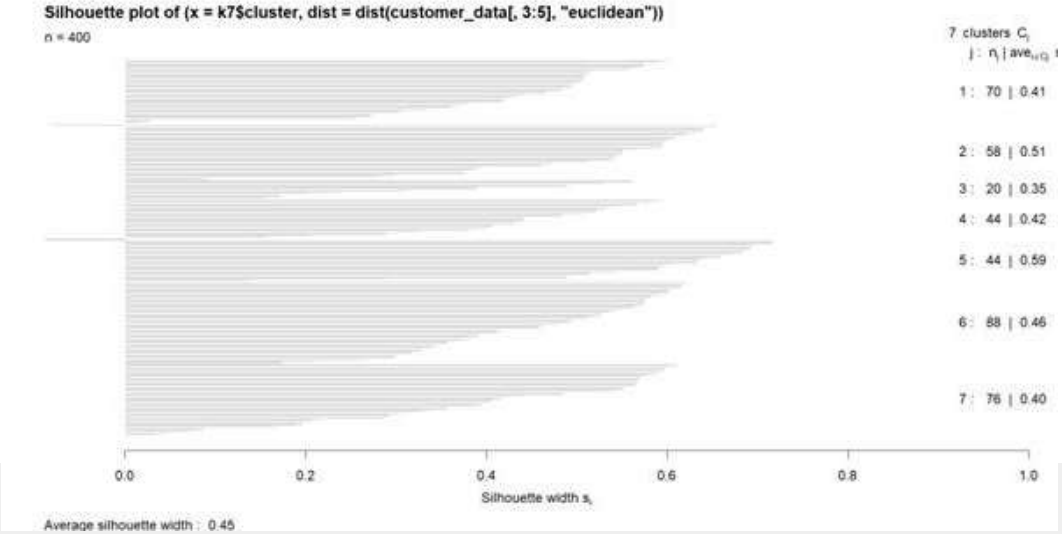


Determining Optimal Number of Clusters

- Silhoutte Plots
  - We did the same for all values from k=2 to 10
  - These are some of the results
  - k=6 is the best

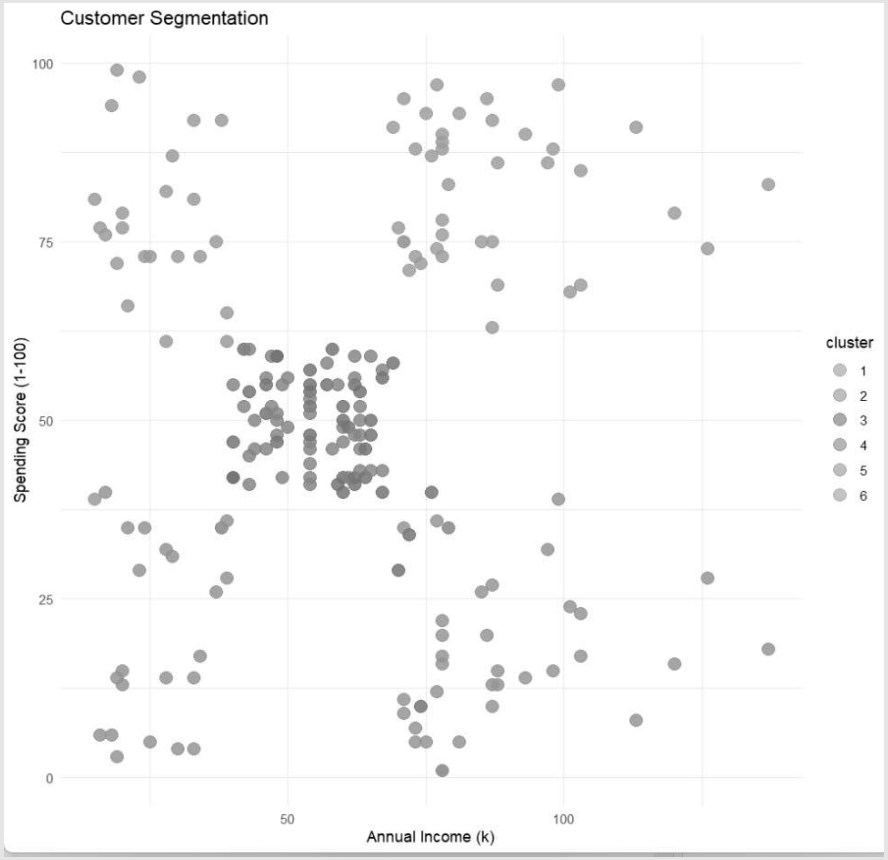






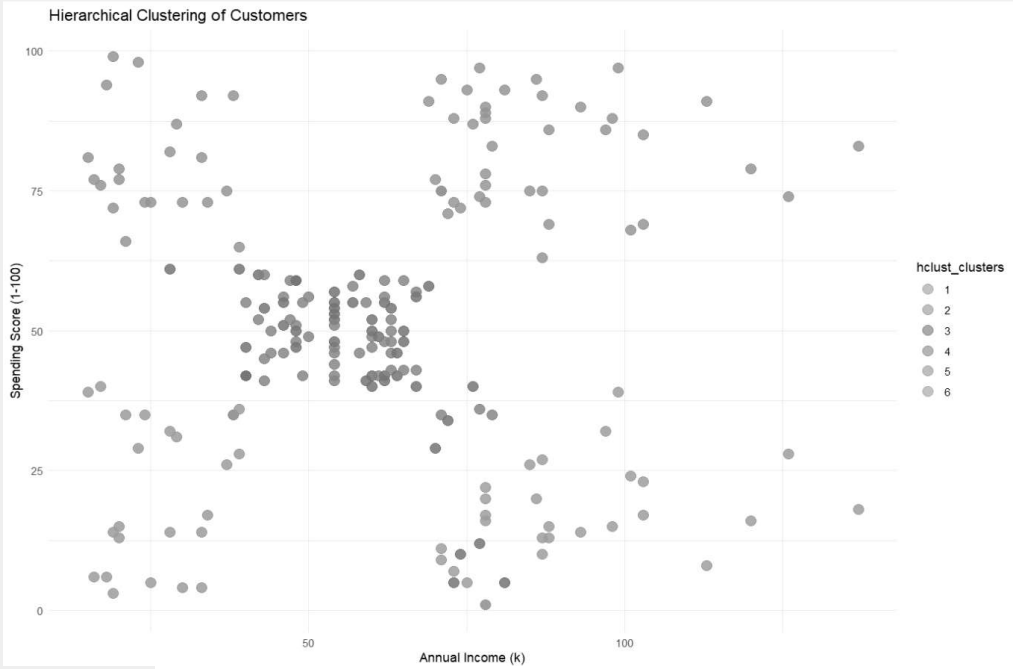
Implementation of Clustering

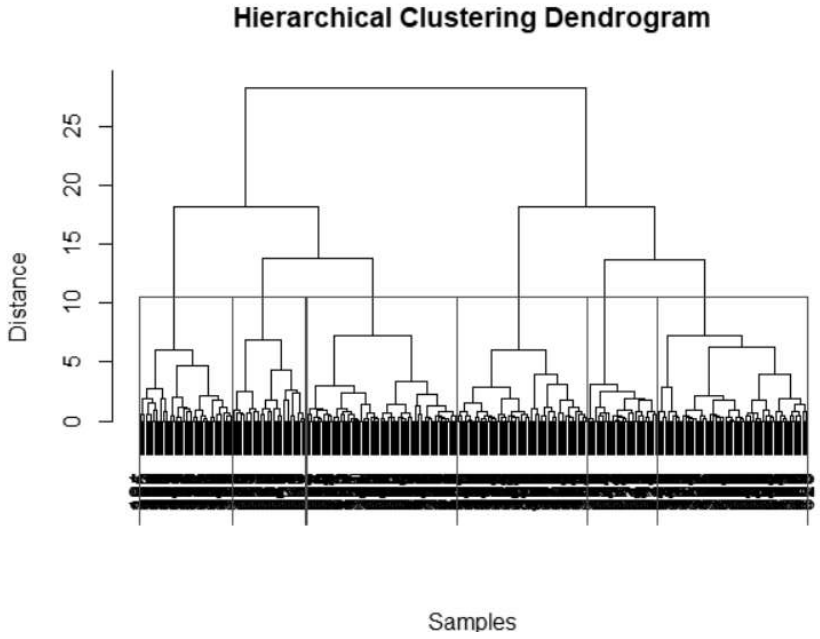
- Using K-Means Clustering
  - We performed K-Means clustering for k=6 to get clusters as such.
  - As we are plotting multiple dimensions in 2-D the clusters are a little overlapping
  - We have chosen Annual Income and Spending Score as the two main variables from PCA.



Implementation of Clustering

- Hierarchical Clustering
  - Clustering using an agglomerative approach to check the validity of the original cluster.
  - Using dendrogram to visualize hierarchy.





Conclusion

- Used 2 clustering technique to group users.
- Visualized the data using EDA techniques
- Found the optimal clusters as k=6
- Got similar clusters for both techniques
- Successfully divided customers into multiple groups