EAS 509: Project - I

2023-04-21

R Markdown

Reading the CSV file into data

```
# Set working directory to the folder containing the CSV file
setwd("C:\\Users\\Sriinitha Reddy\\Downloads")

# Read CSV file into a data frame
data <- read.csv("Exasens.csv", header=TRUE,na.strings = c("", "NA", "N/A"))
```

Before we use the slice function, we need the dplyr package we need to install it and import it

Drop the second and third rows from the data frame as second is empty

and third row consists of min,max which is not required

Preprocessing -> step -1

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Drop the second and third ro two rows from the data frame
data <- slice(data, -(1:2))
```

Viewing the first few rows of the dataset after removing few rows

```
# View the first 10 rows of a data frame called my_data
head(data, n = 10)
```

	Diagnosis <chr>	ID <chr>	Imaginary.Part <chr>	X <chr>	Real.Part <chr>	X.1 <chr>	Gen… <int>	A.. <int>	Smok.. <int>
1	COPD	301-4	-320.61	-300.5635307	-495.26	-464.1719907	1	77	2
2	COPD	302-3	-325.39	-314.7503595	-473.73	-469.2631404	0	72	2
3	COPD	303-3	-323	-317.4360556	-476.12	-471.8976667	1	73	3
4	COPD	304-4	-327.78	-317.3996698	-473.73	-468.856388	1	76	2
5	COPD	305-4	-325.39	-316.1557853	-478.52	-472.8697828	0	65	2
6	COPD	306-3	-327.78	-318.6775535	-507.23	-469.0241943	1	60	2
7	COPD	307-3	-330.18	-320.6174777	-473.73	-467.3618538	1	76	2
8	COPD	308	*NA*	*NA*	*NA*	*NA*	1	77	2
9	COPD	309-4	-320.61	-307.5995856	-476.12	-470.1816328	1	74	2
10	COPD	310-4	-315.82	-300.104765	-473.73	-466.3786343	1	67	2

1-10 of 10 rows

Columns in the dataset

```
# Printing column names
colnames(data)
```

```
## [1] "Diagnosis"      "ID"             "Imaginary.Part" "X"
## [5] "Real.Part"      "X.1"            "Gender"         "Age"
## [9] "Smoking"
```

Checking for NA values in each of the column

```
for (column in colnames(data)) {
  num_missing <- sum(is.na(data[[column]]))
  print(paste0("Number of missing values in ", column, ": ", num_missing))
}
```

```
## [1] "Number of missing values in Diagnosis: 0"
## [1] "Number of missing values in ID: 0"
## [1] "Number of missing values in Imaginary.Part: 299"
## [1] "Number of missing values in X: 299"
## [1] "Number of missing values in Real.Part: 299"
## [1] "Number of missing values in X.1: 299"
## [1] "Number of missing values in Gender: 0"
## [1] "Number of missing values in Age: 0"
## [1] "Number of missing values in Smoking: 0"
```

Preprocessing -> step -2

Replacing NA values with mean value of each column

```
# Convert the Imaginary.Part column to numeric class
data$Imaginary.Part <- as.numeric(data$Imaginary.Part)
data$Real.Part <- as.numeric(data$Real.Part)
data$X <- as.numeric(data$X)
data$X.1 <- as.numeric(data$X.1)
# Calculate the mean of the column and replace missing values with the mean

mean_img <- mean(data$Imaginary.Part, na.rm = TRUE)
data$Imaginary.Part[is.na(data$Imaginary.Part)] <- mean_img
mean_real <- mean(data$Imaginary.Part, na.rm = TRUE)
data$Real.Part[is.na(data$Real.Part)] <- mean_real
mean_X1 <- mean(data$X.1, na.rm = TRUE)
data$X.1[is.na(data$X.1)] <- mean_X1
mean_X <- mean(data$X, na.rm = TRUE)
data$X[is.na(data$X)] <- mean_X
```

dataframe after the step -2 of preprocessing

```
head(data, n = 10)
```

	Diagnosis	ID	Imaginary.Part	X	Real.Part	X.1	Gen…	A..	Smoki…
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>
1	COPD	301-4	-320.6100	-300.5635	-495.2600	-464.1720	1	77	2
2	COPD	302-3	-325.3900	-314.7504	-473.7300	-469.2631	0	72	2

	Diagnosis	ID	Imaginary.Part	X	Real.Part	X.1	Gen…	A..	Smoki…
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>
3	COPD	303-3	-323.0000	-317.4361	-476.1200	-471.8977	1	73	3
4	COPD	304-4	-327.7800	-317.3997	-473.7300	-468.8564	1	76	2
5	COPD	305-4	-325.3900	-316.1558	-478.5200	-472.8698	0	65	2
6	COPD	306-3	-327.7800	-318.6776	-507.2300	-469.0242	1	60	2
7	COPD	307-3	-330.1800	-320.6175	-473.7300	-467.3619	1	76	2
8	COPD	308	-314.9418	-304.7797	-314.9418	-458.7017	1	77	2
9	COPD	309-4	-320.6100	-307.5996	-476.1200	-470.1816	1	74	2
10	COPD	310-4	-315.8200	-300.1048	-473.7300	-466.3786	1	67	2

1-10 of 10 rows

Checking if there are any NA values

```
# Check for NA or null values in the column

for (column in colnames(data)) {
  null_values <- sum(is.na(data[[column]]))
  print(paste0("Number of missing values in ", column, " are: ", null_values))
}
```

```
## [1] "Number of missing values in Diagnosis are: 0"
## [1] "Number of missing values in ID are: 0"
## [1] "Number of missing values in Imaginary.Part are: 0"
## [1] "Number of missing values in X are: 0"
## [1] "Number of missing values in Real.Part are: 0"
## [1] "Number of missing values in X.1 are: 0"
## [1] "Number of missing values in Gender are: 0"
## [1] "Number of missing values in Age are: 0"
## [1] "Number of missing values in Smoking are: 0"
```

Preprocessing -> step -3

Converting Imaginary, Real, X.1, X column values to float type

```
# Convert the values in Imaginary part, Real part to float type
data$Imaginary.Part <- as.numeric(data$Imaginary.Part)
data$Real.Part <- as.numeric(data$Real.Part)
data$X <- as.numeric(data$X)
data$X.1 <- as.numeric(data$X.1)

head(data, n = 10)
```

	Diagnosis	ID	Imaginary.Part	X	Real.Part	X.1	Gen…	A..	Smoki…
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>
1	COPD	301-4	-320.6100	-300.5635	-495.2600	-464.1720	1	77	2
2	COPD	302-3	-325.3900	-314.7504	-473.7300	-469.2631	0	72	2
3	COPD	303-3	-323.0000	-317.4361	-476.1200	-471.8977	1	73	3
4	COPD	304-4	-327.7800	-317.3997	-473.7300	-468.8564	1	76	2
5	COPD	305-4	-325.3900	-316.1558	-478.5200	-472.8698	0	65	2
6	COPD	306-3	-327.7800	-318.6776	-507.2300	-469.0242	1	60	2
7	COPD	307-3	-330.1800	-320.6175	-473.7300	-467.3619	1	76	2
8	COPD	308	-314.9418	-304.7797	-314.9418	-458.7017	1	77	2
9	COPD	309-4	-320.6100	-307.5996	-476.1200	-470.1816	1	74	2
10	COPD	310-4	-315.8200	-300.1048	-473.7300	-466.3786	1	67	2

1-10 of 10 rows

Preprocessing -> step -4

Converting the values to absolute values or ease of visualizing the data and drawing

```
data <- data %>%
  mutate_at(vars(matches("Real.Part|Imaginary.Part|X")), abs)

head(data, n = 10)
```

	Diagnosis	ID	Imaginary.Part	X	Real.Part	X.1	Gen…	A..	Smoki…
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>
1	COPD	301-4	320.6100	300.5635	495.2600	464.1720	1	77	2

	Diagnosis	ID	Imaginary.Part	X	Real.Part	X.1	Gen…	A..	Smoki…
	<chr>	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<int>	<int>	<int>
2	COPD	302-3	325.3900	314.7504	473.7300	469.2631	0	72	2
3	COPD	303-3	323.0000	317.4361	476.1200	471.8977	1	73	3
4	COPD	304-4	327.7800	317.3997	473.7300	468.8564	1	76	2
5	COPD	305-4	325.3900	316.1558	478.5200	472.8698	0	65	2
6	COPD	306-3	327.7800	318.6776	507.2300	469.0242	1	60	2
7	COPD	307-3	330.1800	320.6175	473.7300	467.3619	1	76	2
8	COPD	308	314.9418	304.7797	314.9418	458.7017	1	77	2
9	COPD	309-4	320.6100	307.5996	476.1200	470.1816	1	74	2
10	COPD	310-4	315.8200	300.1048	473.7300	466.3786	1	67	2

1-10 of 10 rows

Preprocessing -> step -5

Removing the ID column as it will not play a significant role.

```
cleaned_data <- select(data, -ID)
#head(data, n = 10)
```

List of columns after removing the ID column

```
colnames(cleaned_data)
```

```
## [1] "Diagnosis"      "Imaginary.Part" "X"              "Real.Part"
## [5] "X.1"            "Gender"         "Age"            "Smoking"
```

VISUALIZATIONS

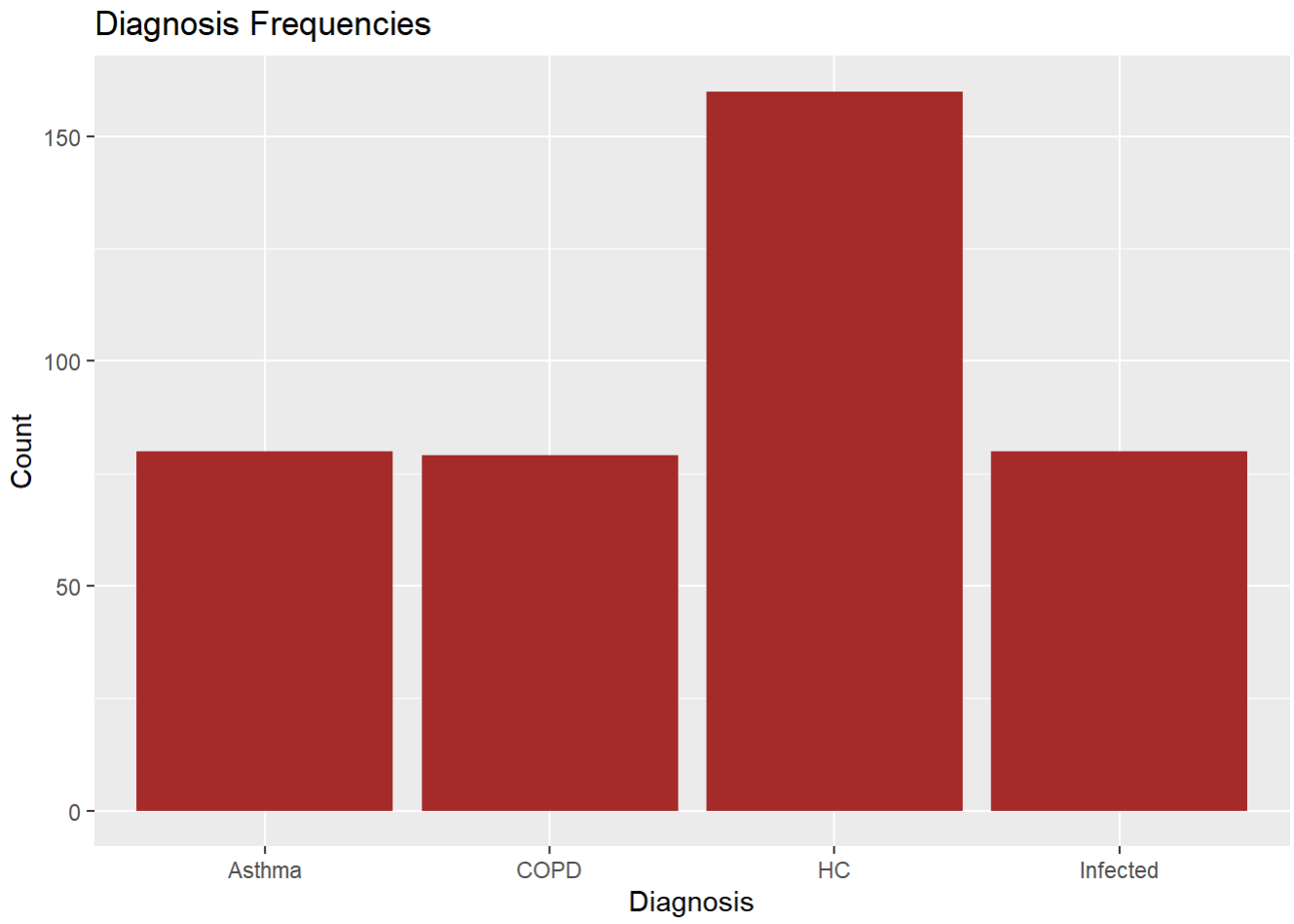
(1). Diagnosis Frequency

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
d_c <- table(cleaned_data$Diagnosis)

d_c_df <- data.frame(Diagnosis = names(d_c), Count = d_c)

ggplot(d_c_df, aes(x = Diagnosis, y = d_c)) +
  geom_bar(stat = "identity",fill = "#A52A2A") +
  labs(title = "Diagnosis Frequencies", x = "Diagnosis", y = "Count")
```

```
## Don't know how to automatically pick scale for object of type <table>.
## Defaulting to continuous.
```



Preprocessing -> step - 6

Performing encoding on the diagnosis column to plot a heat map

```
heatmap_df = cleaned_data
heatmap_df$Diagnosis <- as.numeric(factor(heatmap_df$Diagnosis))
```

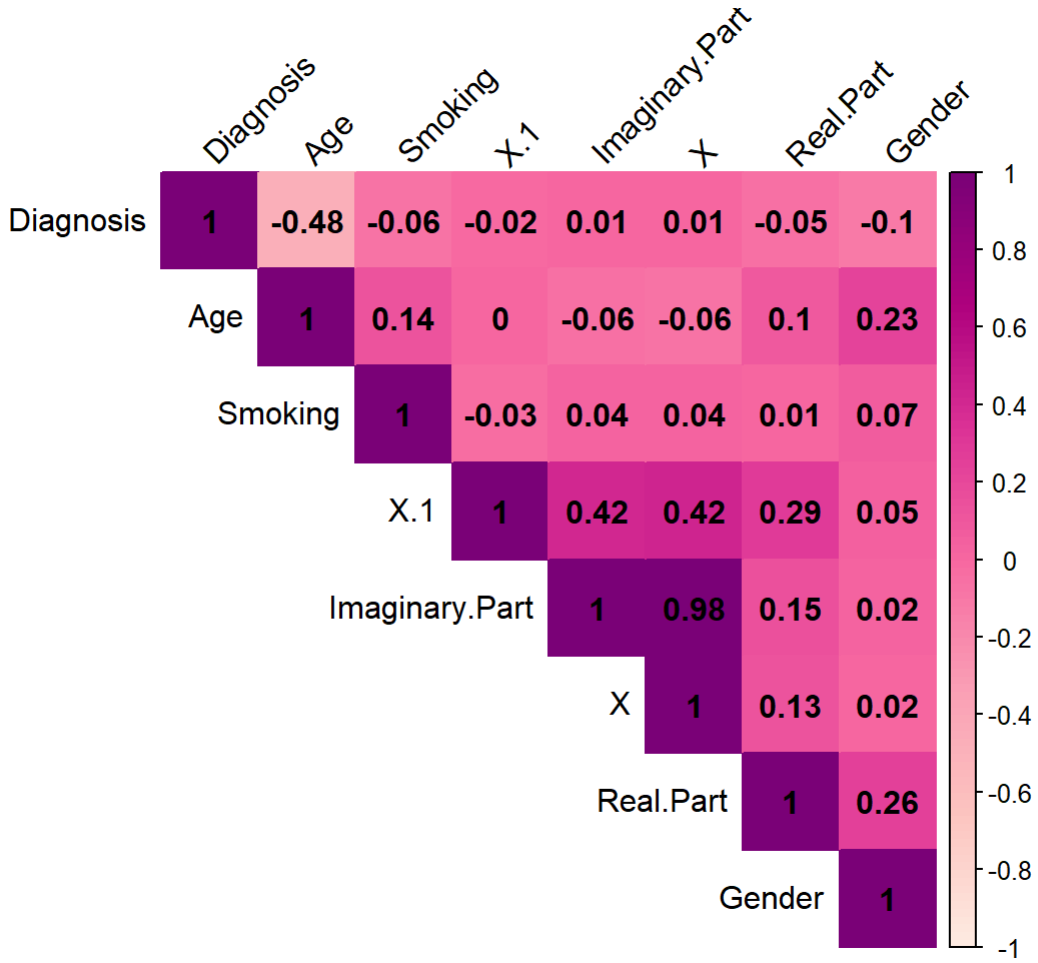
(2). Heatmap

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.2.3
```

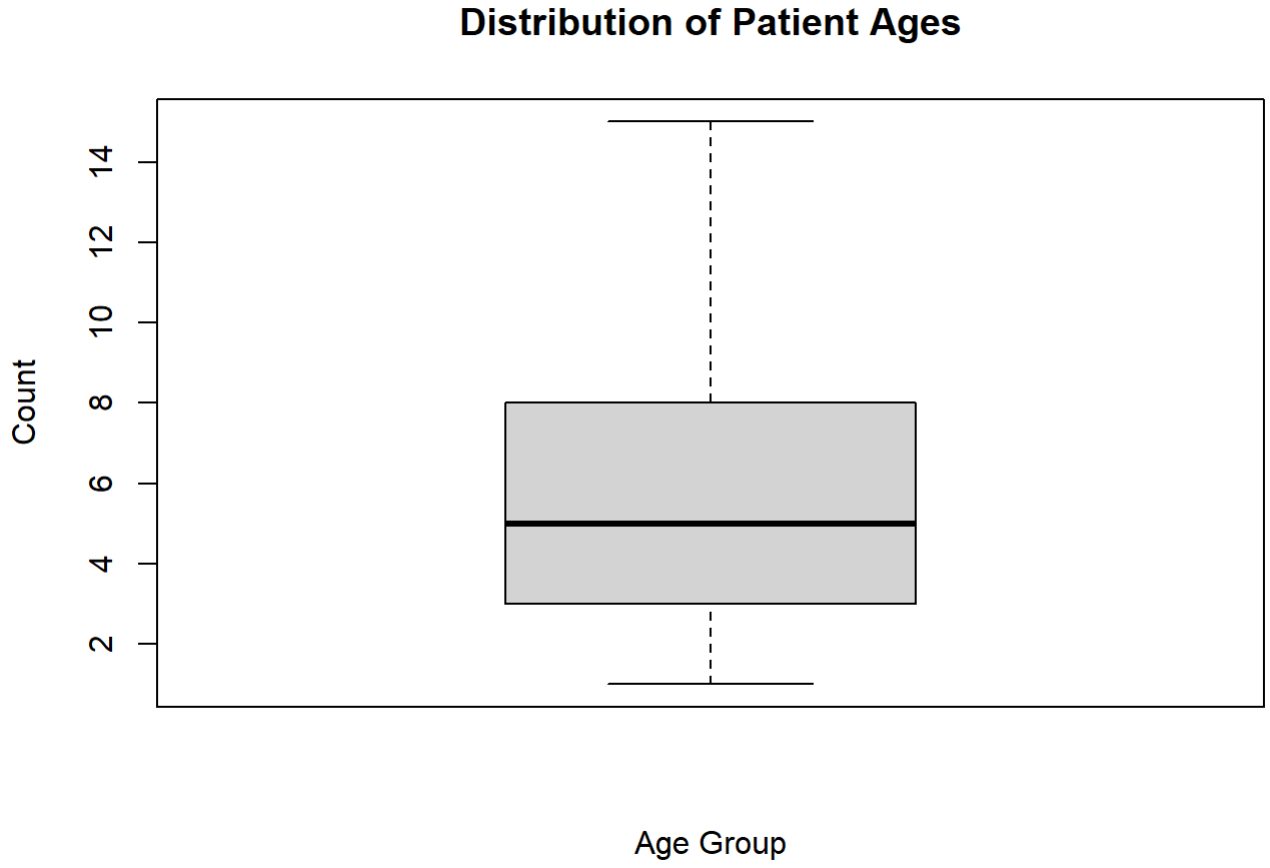
```
## corrplot 0.92 loaded
```

```
# Create a correlation matrix for all variables in the data frame
heatmap <- cor(heatmap_df)

# Plot a heatmap of the correlation matrix
corrplot(heatmap, method = "color", type = "upper", order = "hclust",
         addCoef.col = "black", tl.col = "black", tl.srt = 45,col = colorRampPalette(c("#FEEBE
2", "#FCC5C0", "#FA9FB5", "#F768A1", "#DD3497", "#AE017E", "#7A0177"))(100))
```



(3). Distribution of patient ages

```
# Create a table of the Age column
Age <- table(heatmap_df$Age)

# Create a boxplot of the data with descriptive titles
boxplot(Age,
        main = "Distribution of Patient Ages",
        xlab = "Age Group",
        ylab = "Count")
```



Dropping the X.1 and X column as inferred from the heapmap.

```
# drop columns x2 and x4 from the dataframe
cleaned_data <- cleaned_data[, c(-3, -5)]
print(colnames(cleaned_data))
```

```
## [1] "Diagnosis"      "Imaginary.Part" "Real.Part"      "Gender"
## [5] "Age"            "Smoking"
```

```
#print(class(heatmap_df))
```

Preprocessing -> step -7

Scaling

```
library(stats)

features <- heatmap_df[,c("Diagnosis", "Imaginary.Part", "Real.Part", "Gender", "Age", "Smokin
g")]

# Standardize(scaling) the features
features <- scale(features)
features <- as.data.frame(features)
head(features, n = 10)
```

	Diagnosis <dbl>	Imaginary.Part <dbl>	Real.Part <dbl>	Gender <dbl>	Age <dbl>	Smoking <dbl>
1	-0.5886019	0.39985540	1.9324622	1.2270497	1.5222284	0.3680883
2	-0.5886019	0.73705395	1.6367642	-0.8129204	1.2529329	0.3680883
3	-0.5886019	0.56845468	1.6695890	1.2270497	1.3067920	1.7154941
4	-0.5886019	0.90565323	1.6367642	1.2270497	1.4683693	0.3680883
5	-0.5886019	0.73705395	1.7025511	-0.8129204	0.8759192	0.3680883
6	-0.5886019	0.90565323	2.0968609	1.2270497	0.6066236	0.3680883
7	-0.5886019	1.07495794	1.6367642	1.2270497	1.4683693	0.3680883
8	-0.5886019	0.00000000	-0.5440697	1.2270497	1.5222284	0.3680883
9	-0.5886019	0.39985540	1.6695890	1.2270497	1.3606511	0.3680883
10	-0.5886019	0.06195142	1.6367642	1.2270497	0.9836374	0.3680883

1-10 of 10 rows

Applying the clustering algorithms

1. K-means clustering

Choosing the number of clusters using elbow method

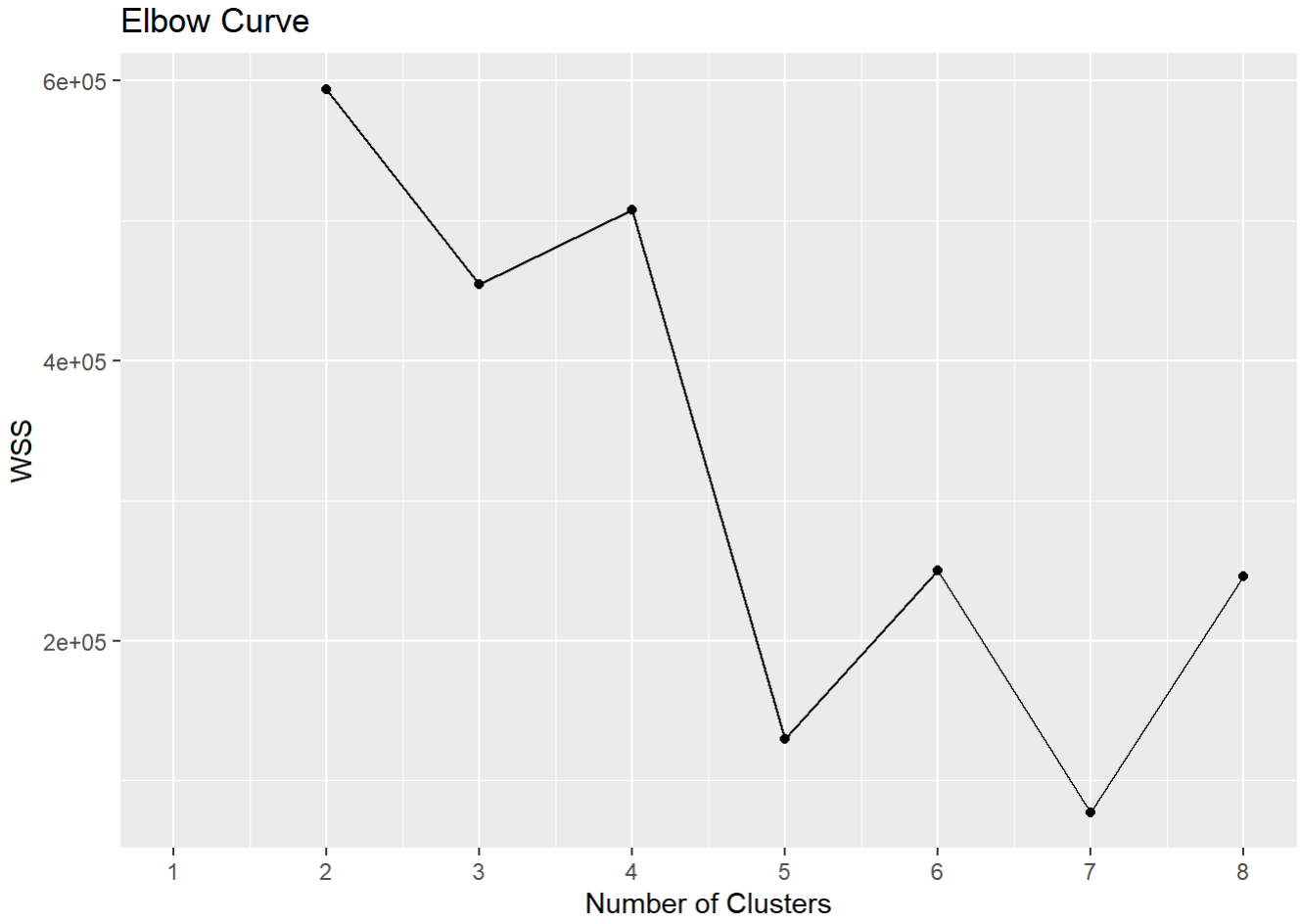
```
library(cluster)
library(ggplot2)

# Compute the within-cluster sum of squares for different values of k
wss <- (nrow(heatmap_df)-1)*sum(apply(data,2,var))
```

```
## Warning in FUN(newX[, i], ...): NAs introduced by coercion

## Warning in FUN(newX[, i], ...): NAs introduced by coercion
```

```
for (i in 2:10)
  wss[i] <- sum(kmeans(heatmap_df, centers=i)$withinss)

# Plot the elbow curve
ggplot(data.frame(x=1:8, y=wss[1:8]), aes(x=x, y=y)) + geom_line() + geom_point() + scale_x_cont
inuous(breaks=1:8) +labs(x="Number of Clusters", y="WSS") +ggtitle("Elbow Curve")
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_point()`).
```



From the above elbow plot, we can infer that the bend occurs when the number of clusters are 3, hence we choose

the number of clusters as 3.

```
# Perform k-means clustering
k_best = 3
kmeans_model <- kmeans(features, k_best)
```

```
# View the clustering results
print(kmeans_model)
```

```
## K-means clustering with 3 clusters of sizes 109, 205, 85
##
## Cluster means:
##    Diagnosis Imaginary.Part   Real.Part     Gender        Age    Smoking
## 1 -0.5526918    -0.05694916  0.42297837  1.2270497  0.7780834  0.3804498
## 2  0.7670488    -0.02651449 -0.20384272 -0.3153667 -0.6145331 -0.2628920
## 3 -1.1411953     0.13697565 -0.05078688 -0.8129204  0.4843318  0.1461626
##
## Clustering vector:
##   [1] 1 3 1 1 3 1 1 1 1 1 1 1 3 1 3 1 1 1 1 1 1 1 1 1 3 3 1 1 1 1 1 1 1 3 1 3
##  [38] 1 3 1 1 1 3 1 3 1 1 1 1 3 1 1 3 3 3 1 1 3 1 3 1 1 1 1 1 3 1 1 3 1 1 3 1 1
##  [75] 3 3 3 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 1 2 2 2 2 1 2 2 1 2 2 1 2 2 2
## [112] 2 2 2 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 1 2 2 1 2 1 2 3 1 1 1 1 2 1
## [149] 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 1 1 2 3 2 2 2 2 2 2 2 2 1 2
## [186] 2 2 2 2 2 2 2 1 1 3 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2
## [223] 1 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 3 3 3 3 1 3 3 3 3 3 3 3 3 3 3 3 1 3 1
## [260] 3 3 3 3 3 3 3 3 3 1 3 3 1 1 3 3 3 1 1 1 3 3 1 3 3 3 3 1 3 3 3 3 1 3 1 1 1
## [297] 3 1 3 1 1 1 3 1 1 3 3 1 3 3 3 3 3 1 3 1 1 3 3 2 2 2 2 2 2 2 1 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [371] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##
## Within cluster sum of squares by cluster:
## [1] 577.0112 811.5150 268.7500
##  (between_SS / total_SS =  30.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Plotting the clusters

```
# Plotting the clusters
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.2.3
```

```
autoplot(kmeans_model,features,frame=TRUE)+
  scale_fill_manual(values = c("red", "green", "blue")) +
  labs(fill = "Cluster")
```

HYPERPARAMETER TUNING (Varying the value of K)

```
k_best = 4
kmeans_model_1 <- kmeans(features, k_best)
library(ggfortify)
autoplot(kmeans_model_1,features,frame=TRUE)+
  scale_fill_manual(values = c("red", "green", "blue","orange")) +
  labs(fill = "Cluster")
```


```
k_best = 6
kmeans_model_2 <- kmeans(features, k_best)
library(ggfortify)
autoplot(kmeans_model_2,features,frame=TRUE)+
  scale_fill_manual(values = c("red", "green", "blue","black","orange","yellow")) +
  labs(fill = "Cluster")
```

EVALUATION METRIC - SILHOUETTE SCORE - K.MEANS CLUSTERING

```
knn_score <- silhouette(kmeans_model$cluster, dist(heatmap_df))

# Calculate the average Silhouette score
avg_score <- mean(knn_score[, 3])

# Print the average Silhouette score
print(paste("Average Silhouette score for the K-means clustering is:", avg_score))
```

```
## [1] "Average Silhouette score for the K-means clustering is: 0.0284360445562191"
```

2. Hierarchical Clustering

Performing clustering using all the 4 methods

```
distance <- dist(heatmap_df, method = "euclidean")
complete_h <- hclust(distance, method = "complete")
single_h <- hclust(distance,method = "single")
avg_h <- hclust(distance,method = "average")
centroid_h <- hclust(distance,method="centroid")
```

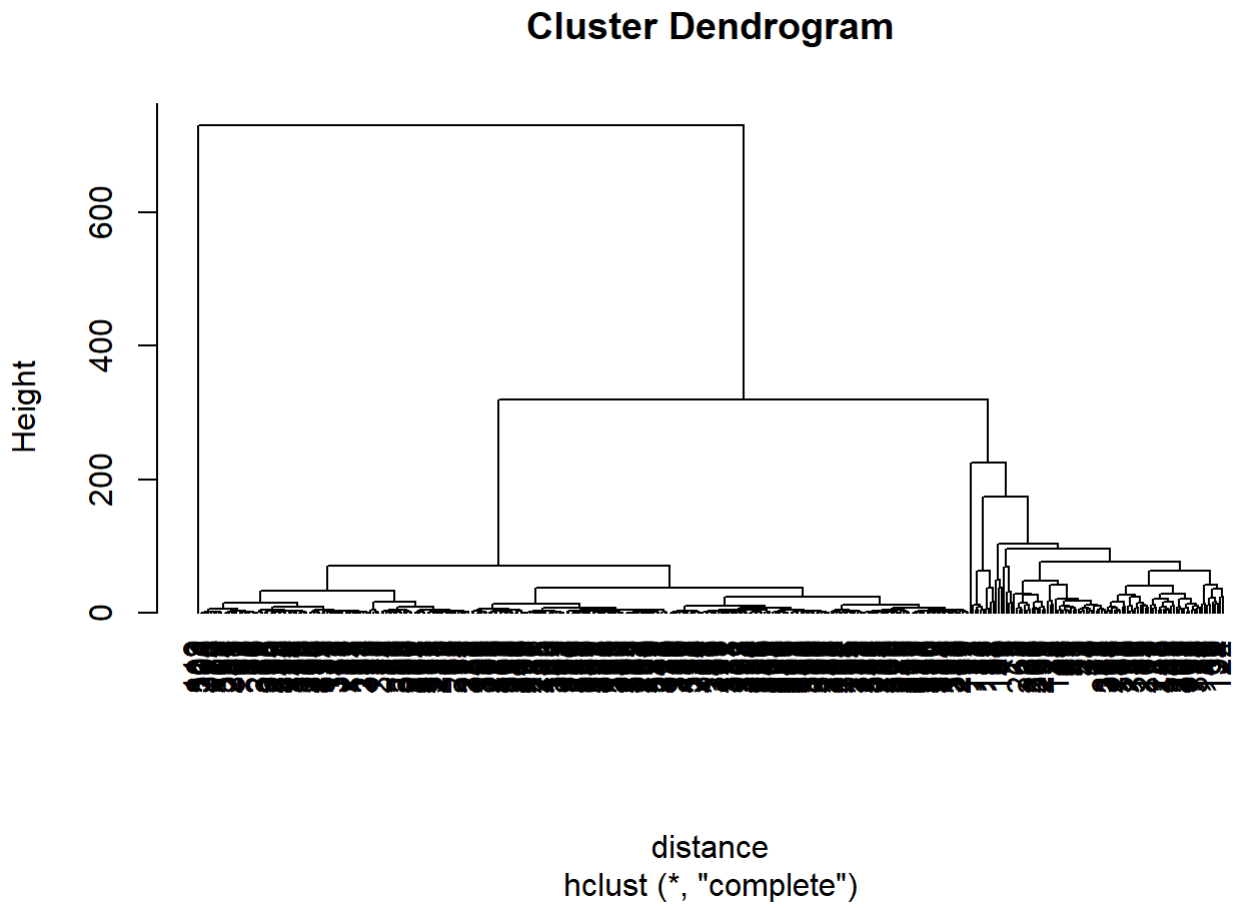
Plot dendrograms for all the methods

Dendrogram - Complete method

```
cutree(complete_h,4)
```

```
##   [1] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 2 1 2
##  [38] 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 3 1 2 1 2 2 2 2 1 2 2 2 2 1 2
## [112] 2 1 1 1 1 1 2 4 2 2 1 1 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 1 1 2 1 1 1 1 2 1 1
## [149] 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1
## [260] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
## [371] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
plot(complete_h, hang = -1)
```

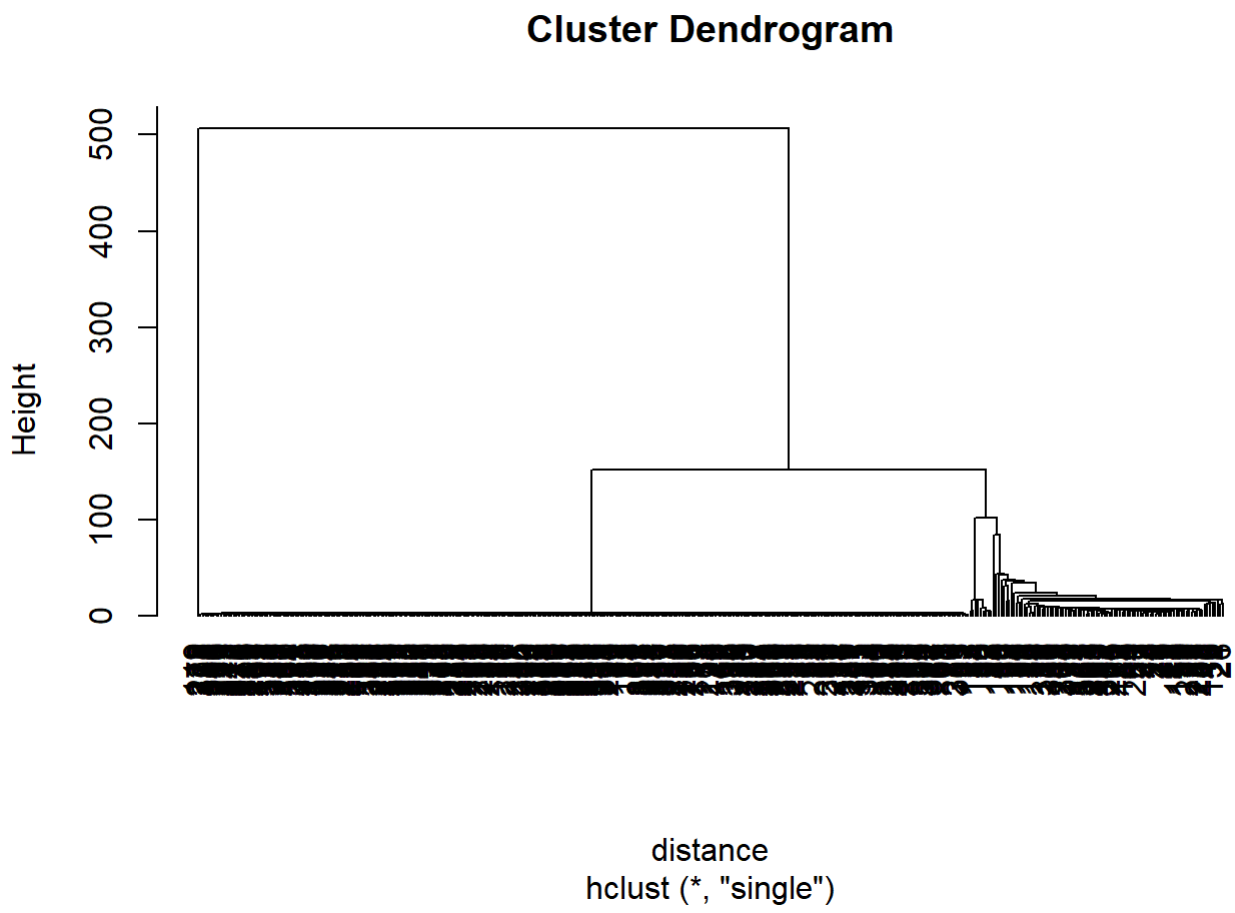


Dendrogram - single method

```
cutree(single_h,4)
```

```
##   [1] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 3 3 2 1 1 1 1 1 2 2 3 3 1 2 3 2
##  [38] 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 1 1 2 1 2 2 2 2 1 2 2 2 2 1 2
## [112] 2 1 1 1 1 1 2 4 2 2 1 3 2 2 1 2 2 2 1 3 3 2 3 2 2 2 1 1 1 2 1 1 1 1 2 1 1
## [149] 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1
## [260] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1
## [371] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
plot(single_h, hang = -1)
```

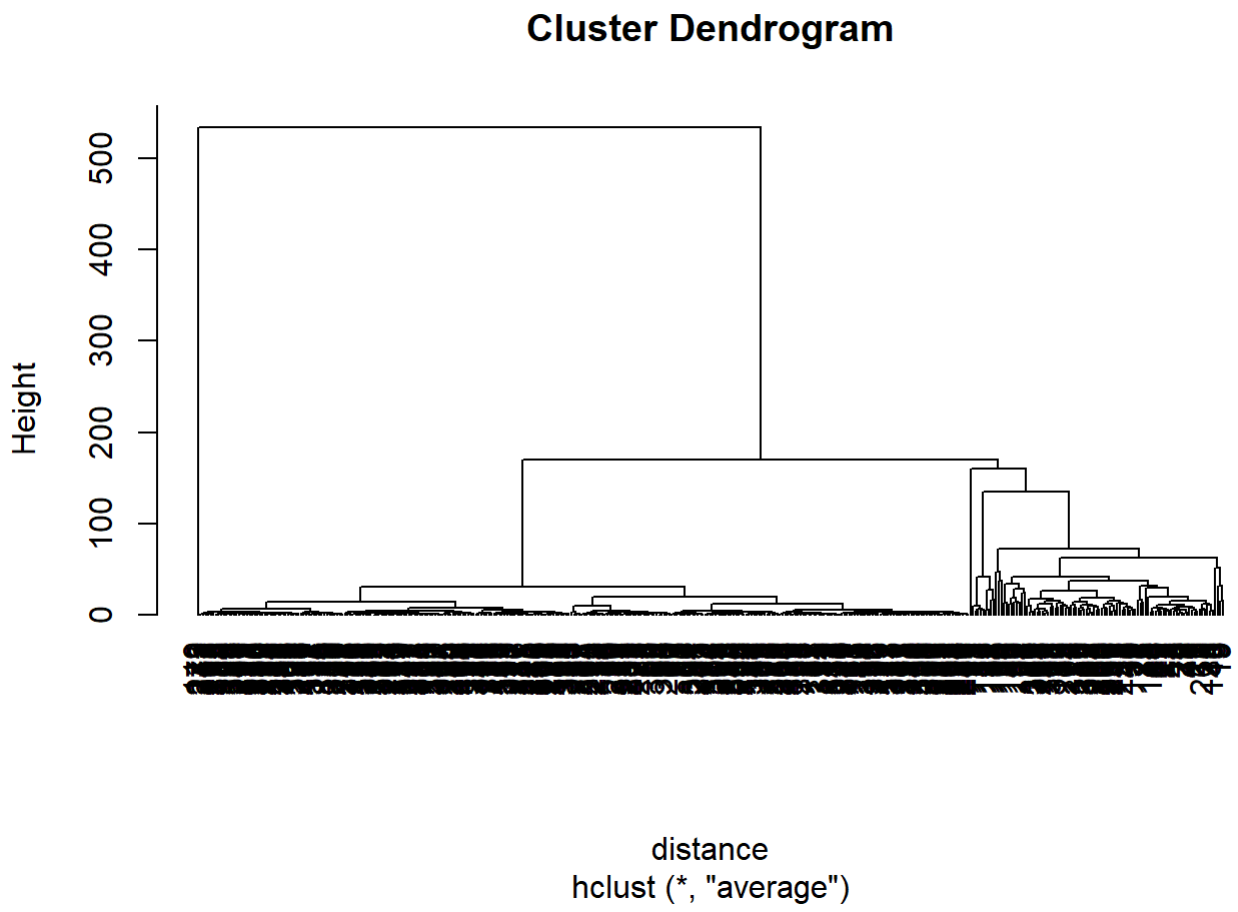


Dendrogram - average method

```
cutree(avg_h,4)
```

```
##   [1] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 2 1 2
##  [38] 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 3 1 2 1 2 2 2 2 1 2 2 2 2 1 2
## [112] 2 1 1 1 1 1 2 4 2 2 1 1 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 1 1 2 1 1 1 1 2 1 1
## [149] 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1
## [260] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1
## [371] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
plot(avg_h, hang = -1)
```

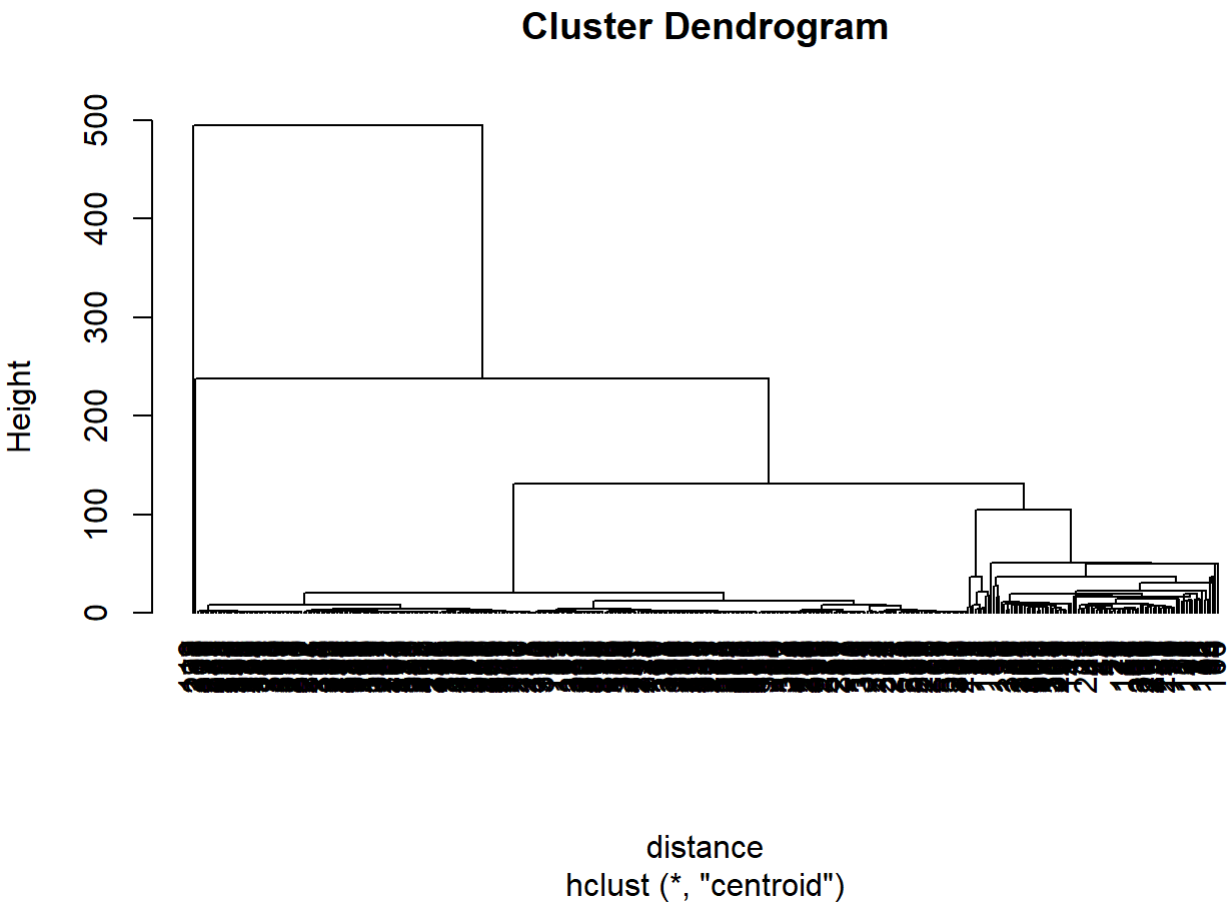


Dendrogram - centroid method

```
cutree(centroid_h,4)
```

```
##   [1] 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 2 2 1 1 1 1 1 1 2 1 1 1 1 1 2 2 1 1 1 2 1 2
##  [38] 1 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [75] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 3 1 2 1 2 2 2 2 1 2 2 2 2 1 2
## [112] 2 1 1 1 1 1 2 4 2 2 1 1 2 2 1 2 2 2 1 1 1 2 1 2 2 2 1 1 1 2 1 1 1 1 2 1 1
## [149] 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [186] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [223] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1
## [260] 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [297] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [334] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1
## [371] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
plot(centroid_h, hang = -1)
```



EVALUATION METRIC - SILHOUETTE SCORE - HEIRARCHICAL CLUSTERING (all 4 methods)

Silhouette score: Method 1- Complete Method

```
# The number of clusters we choose here are 3 for hierarchical clustering
complete_score <- silhouette(cutree(complete_h, k = 3), dist(heatmap_df))
avg_score_complete <- mean(complete_score[, 3])
print(paste("Average Silhouette score using complete method is:",avg_score_complete))
```

```
## [1] "Average Silhouette score using complete method is: 0.826030750636403"
```

Silhouette score: Method 2 - Single Method

```
single_score <- silhouette(cutree(single_h, k = 3), dist(heatmap_df))
avg_score_single <- mean(single_score[, 3])
print(paste("Average Silhouette score using single method is:",avg_score_single))
```

```
## [1] "Average Silhouette score using single method is: 0.826030750636403"
```

Silhouette score: Method 3 - Average Method

```
average_score <- silhouette(cutree(avg_h, k = 3), dist(heatmap_df))
avg_score_method <- mean(average_score[, 3])
print(paste("Average Silhouette score using average method is:",avg_score_method))
```

```
## [1] "Average Silhouette score using average method is: 0.826030750636403"
```

Silhouette score: Method 4 - Centroid Method

```
centroid_score <- silhouette(cutree(centroid_h, k = 3), dist(heatmap_df))
avg_score_centroid <- mean(centroid_score[, 3])
print(paste("Average Silhouette score using centroid method is:",avg_score_centroid))
```

```
## [1] "Average Silhouette score using centroid method is: 0.642083063167706"
```