Project-1

2024-04-13

#loading the dataset

```
dataset <- read.csv('C:\\Users\\Deepak\\Downloads\\Exasens.csv')
#dataset is loaded and assigns to a dataset variable
```

#Prints the top 5 rows

```
head(dataset, 5)
```

```
##   Diagnosis    ID Imaginary.Part            X Real.Part          X.1 Gender Age
## 1                                                                          NA  NA
## 2                          Min         Avg.       Min         Avg.      NA  NA
## 3      COPD 301-4        -320.61 -300.5635307   -495.26 -464.1719907       1  77
## 4      COPD 302-3        -325.39 -314.7503595   -473.73 -469.2631404       0  72
## 5      COPD 303-3           -323 -317.4360556   -476.12 -471.8976667       1  73
##   Smoking X.2 X.3      X.4             X.5
## 1      NA  NA  NA
## 2      NA  NA  NA  Gender          Somking
## 3       2  NA  NA   Male=1    Non-smoker=1
## 4       2  NA  NA Female=0     Ex-smoker=2
## 5       3  NA  NA          Active-smoker=3
```

dim is used to find the dimensions of the dataset

```
dim(dataset)
```

```
## [1] 401  13
```

#summary function is used to defines the statistical information of each column

```
summary(dataset)
```

```
##   Diagnosis              ID            Imaginary.Part          X
##  Length:401         Length:401         Length:401         Length:401
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
##
##   Real.Part             X.1                Gender            Age
##  Length:401         Length:401         Min.   :0.0000   Min.   :17.00
##  Class :character   Class :character   1st Qu.:0.0000   1st Qu.:31.00
##  Mode  :character   Mode  :character   Median :0.0000   Median :49.00
##                                        Mean   :0.3985   Mean   :48.74
##                                        3rd Qu.:1.0000   3rd Qu.:64.00
##                                        Max.   :1.0000   Max.   :93.00
##                                        NA's   :2        NA's   :2
##     Smoking        X.2            X.3              X.4
##  Min.   :1.000   Mode:logical   Mode:logical   Length:401
##  1st Qu.:1.000   NA's:401       NA's:401       Class :character
##  Median :2.000                                 Mode  :character
##  Mean   :1.727
##  3rd Qu.:2.000
##  Max.   :3.000
##  NA's   :2
##      X.5
##  Length:401
##  Class :character
##  Mode  :character
##
##
##
##
```

Converting character data type to Numeric data type

```
dataset$Imaginary.Part <- as.numeric(dataset$Imaginary.Part)
```

```
## Warning: NAs introduced by coercion
```

```
dataset$X <- as.numeric(dataset$X)
```

```
## Warning: NAs introduced by coercion
```

```
dataset$Real.Part <- as.numeric(dataset$Real.Part)
```

```
## Warning: NAs introduced by coercion
```

```
dataset$X.1 <- as.numeric(dataset$X.1)
```

```
## Warning: NAs introduced by coercion
```

Replacing the "Imaginary.Part" column with its mean value

```r
mean_val <- mean(dataset$Imaginary.Part, na.rm = TRUE)
dataset$Imaginary.Part <- ifelse(is.na(dataset$Imaginary.Part), mean_val, dataset$Imaginary.Part)
```

Replacing the "Imaginary.Part(X)" column with its mean value

```r
mean_val <- mean(dataset$X, na.rm = TRUE)
dataset$X <- ifelse(is.na(dataset$X), mean_val, dataset$X)
```

Replacing the "Real.Part" column with its mean value

```r
mean_val <- mean(dataset$Real.Part, na.rm = TRUE)
dataset$Real.Part <- ifelse(is.na(dataset$Real.Part), mean_val, dataset$Real.Part)
```

Replacing the "Real.Part(X.1)" column with its mean value

```r
mean_val <- mean(dataset$X.1, na.rm = TRUE)
dataset$X.1 <- ifelse(is.na(dataset$X.1), mean_val, dataset$X.1)
```

Removing unneccessary columns in the dataset that are not usefull for the prediction

```r
dataset <- subset(dataset, select = -c(X.2, X.3,X.4,X.5))
dataset <- subset(dataset,select = -c(ID))
#Removing ID because it has no impact on the diagnosis target variable - Feature selection
```

Counts the number of null values

```r
colSums(is.na(dataset))
```

```
##      Diagnosis Imaginary.Part              X      Real.Part            X.1
##              0              0              0              0              0
##         Gender            Age        Smoking
##              2              2              2
```

After examine the dataset, we notice two instances where the 'smoking' column is empty. Consequently, we opt to eliminate rows from the dataset that have missing values in the 'smoking' column.

```
# Removing the rows that contains smoking as null
dataset <- dataset[complete.cases(dataset$Smoking), ]
```

```
colSums(is.na(dataset))
```

```
##      Diagnosis Imaginary.Part              X      Real.Part            X.1
##              0              0              0              0              0
##         Gender            Age        Smoking
##              0              0              0
```

```
# Finally, To ensure there are no null values
```

Checking for the duplicates

```
sum_duplicated <- sum(duplicated(dataset))
# Prints the sum of duplicated rows
print(sum_duplicated)
```

```
## [1] 61
```

Removing the duplicates from the dataset

```
dataset <- unique(dataset)
```

Scaling of dataset for better accuracy

```
# Selects the columns we want to scale
scale_column <- dataset[, c("Imaginary.Part", "X", "Real.Part", "X.1")]
# Scaled the selected columns
scaled_columns <- scale(scale_column)
# Replaces the original columns with the scaled values
dataset[, c("Imaginary.Part", "X", "Real.Part", "X.1")] <- scaled_columns
```

sapply is used to ensure that neccessary columns are in a numeric data type

```
column_dtypes <- sapply(dataset, class)
print(column_dtypes)
```

```
##      Diagnosis Imaginary.Part              X      Real.Part            X.1
##    "character"      "numeric"      "numeric"      "numeric"      "numeric"
##         Gender            Age        Smoking
##      "integer"      "integer"      "integer"
```

We can see that Diagnosis column are in character type in order to convert into a numeric we performed a labled encoding

```
# Convert the column to a factor
dataset$Diagnosis <- as.numeric(as.factor(dataset$Diagnosis))
```

Exploratory Data Analysis

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
##
## 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
```

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.3.3
```

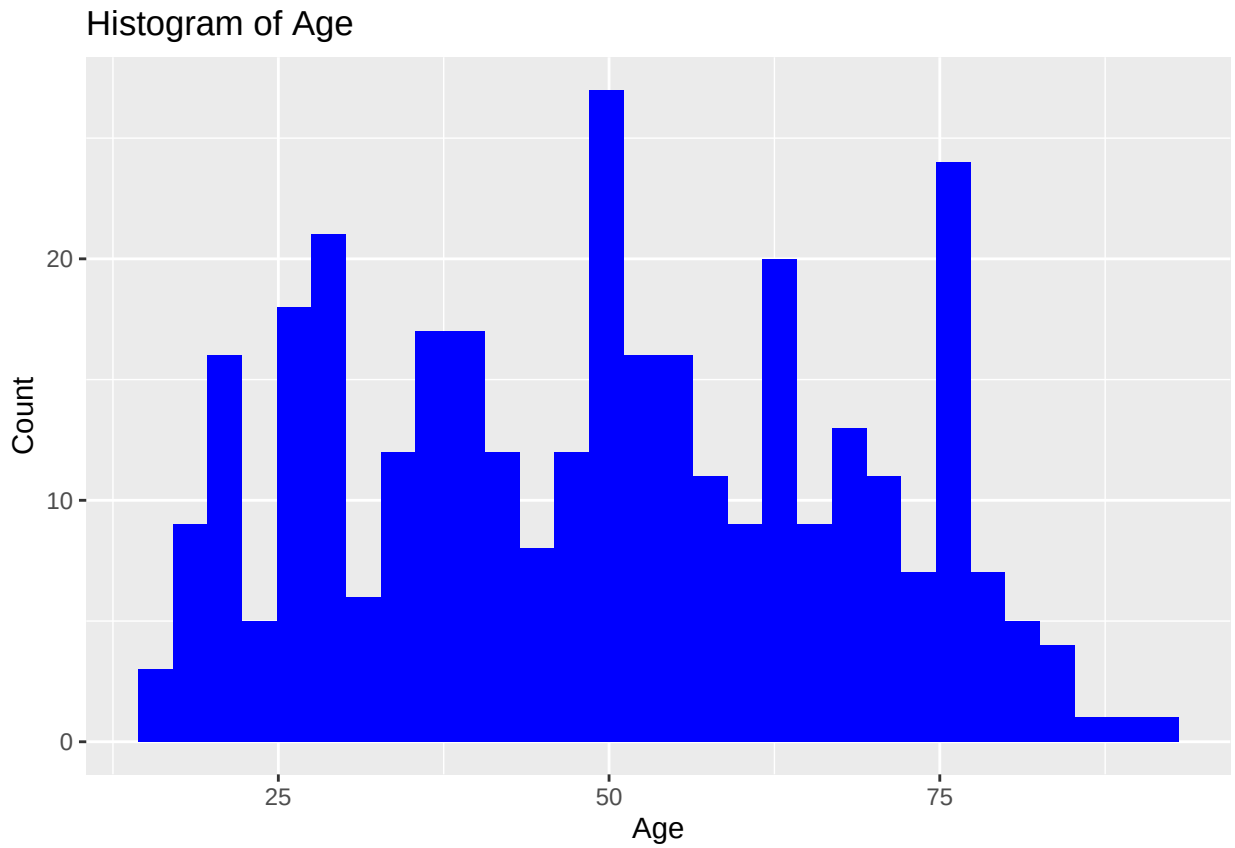
Before proceeding to analysis we check to ensure no column is in character type

```
column_types <- sapply(dataset, class)
print(column_types)
```

```
##      Diagnosis Imaginary.Part              X      Real.Part            X.1
##      "numeric"      "numeric"      "numeric"      "numeric"      "numeric"
##         Gender            Age        Smoking
##      "integer"      "integer"      "integer"
```

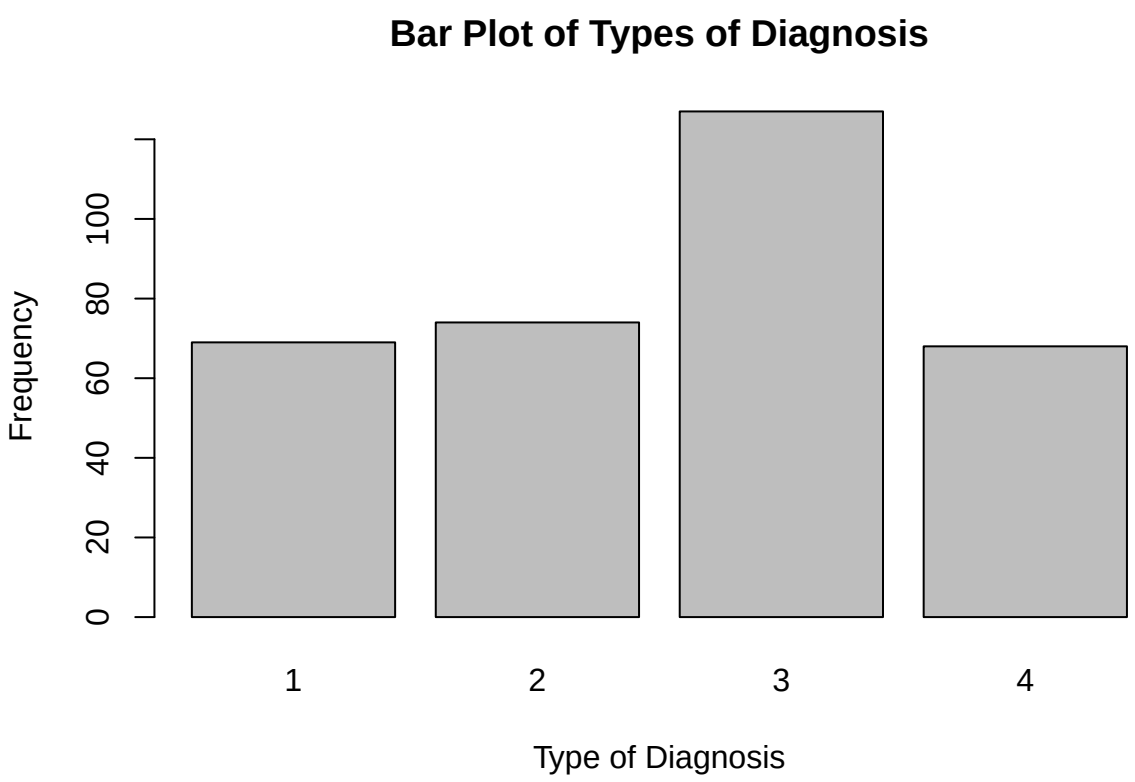
#Histogram #Below code examines the count of different ages across the dataset

```
ggplot(dataset, aes(x = Age)) +
  geom_histogram(bins = 30, fill = "blue") +
  labs(x = "Age", y = "Count", title = "Histogram of Age")
```



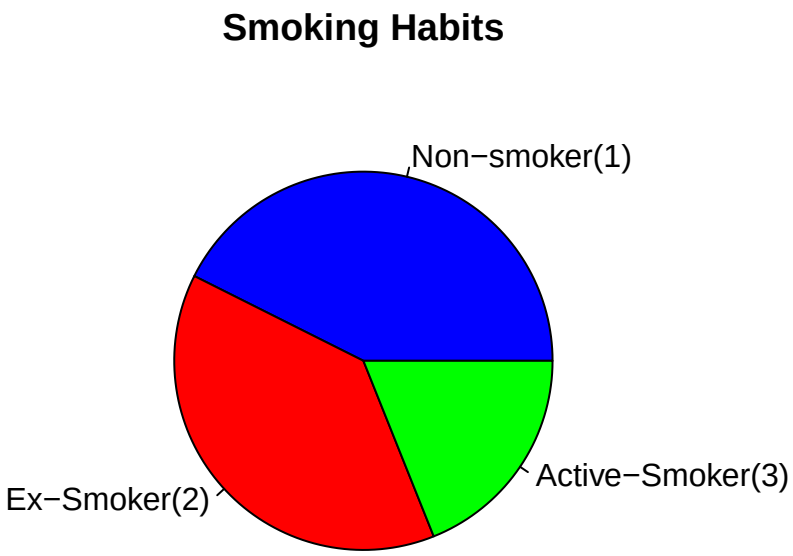
The above histogram displaying the distribution of age in a dataset. The x-axis represents age, divided into bins, while the y-axis shows the count of observations within each age bin. There are notable peaks at various points, suggesting concentrations of individuals at certain ages.

```
counts <- table(dataset$Diagnosis)
# Creates a bar plot
barplot(counts, main = "Bar Plot of Types of Diagnosis", xlab = "Type of Diagnosis", ylab = "Frequency")
```



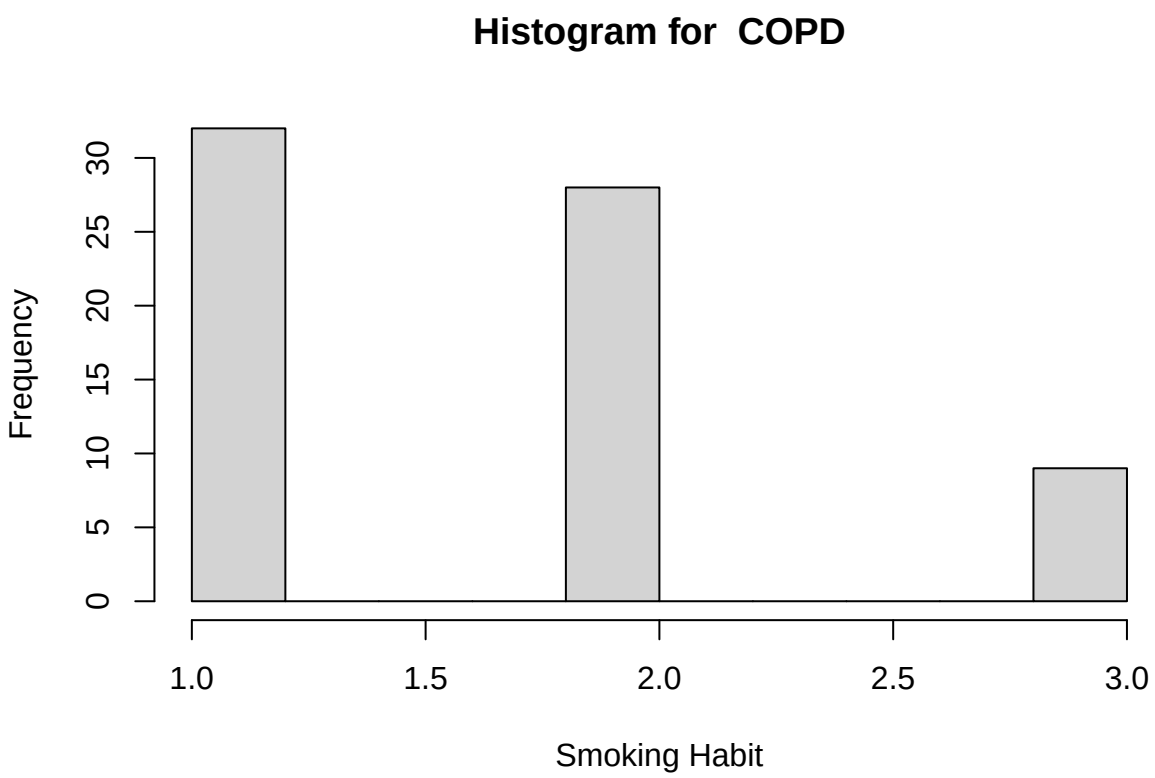
The above bar plot illustrates the frequency of four different types of diagnoses in a dataset. The x-axis categorizes the diagnoses by type, labeled 1 through 4, while the y-axis indicates the frequency of each diagnosis. Type 3 has the highest frequency, suggesting it's the most common diagnosis among the data sampled.

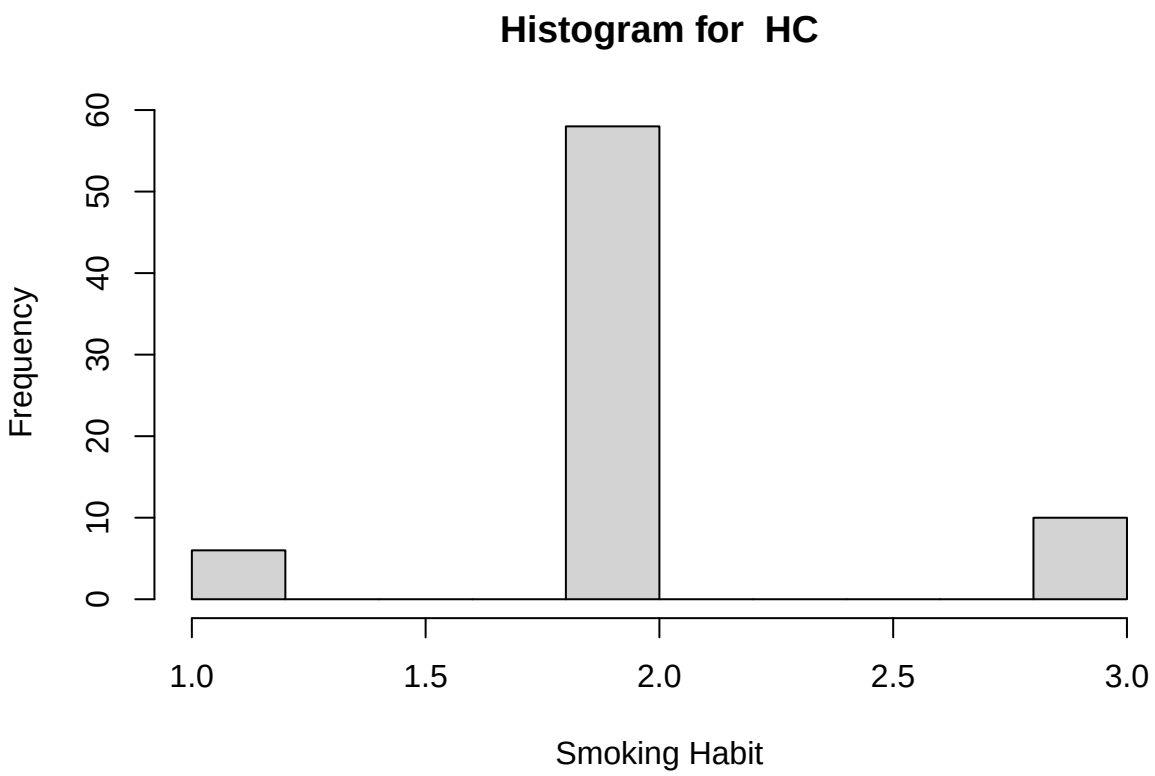
```r
# create a table of counts for each category
counts <- table(dataset$Smoking)
pie(counts,
    labels = c("Non-smoker(1)", "Ex-Smoker(2)", "Active-Smoker(3)"),
    col = c("blue", "red", "green"),
    main = "Smoking Habits")
```

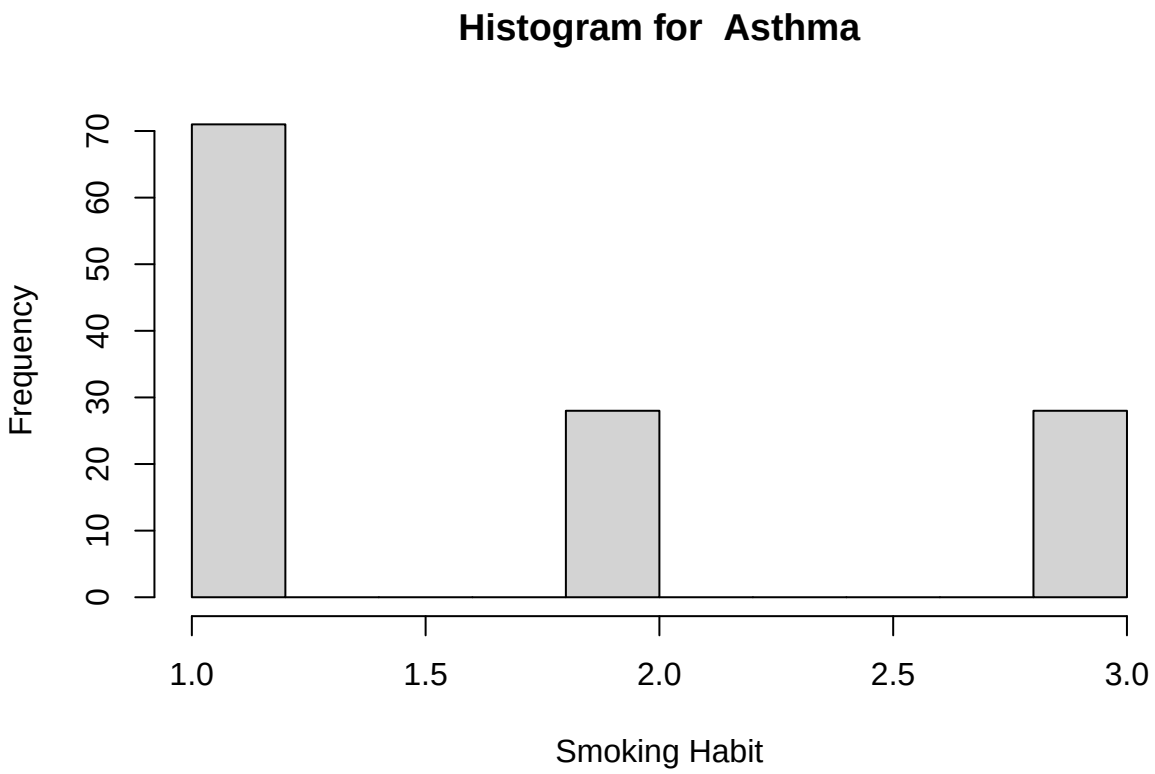


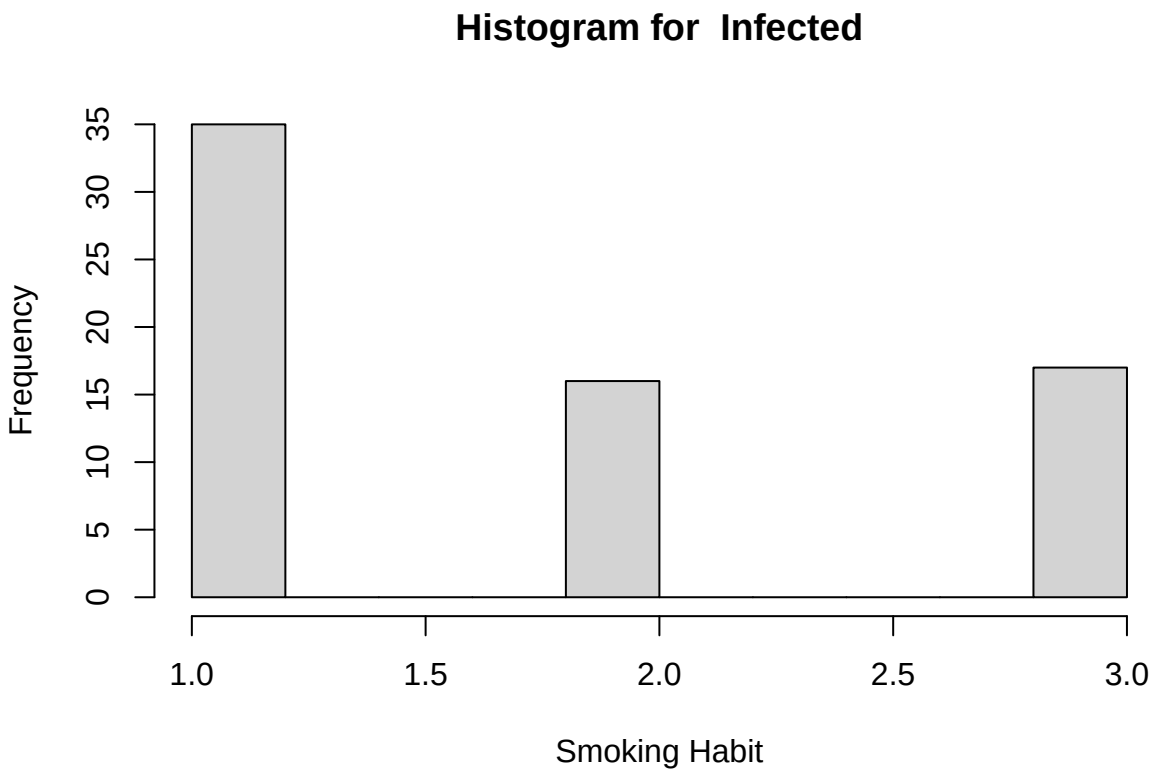
The above pie chart represents the distribution of smoking habits among a group of individuals. The chart is divided into three categories: non-smokers (blue), ex-smokers (red), and active smokers (green). Non-smokers make up the largest segment, indicating they are the majority in this group. # From above it is clear that Non-Smoker has majority among the dataset

```
list <- c("COPD","HC","Asthma","Infected")
# Creates separate histograms for each value of the target variable
for (i in 1:4) {
  # Subset data for the each type of diagnosis
  subset_data <- dataset[dataset$Diagnosis == i, ]
  # Creates a histogram for smoking habit for each type of Diagnosis
  hist(subset_data$Smoking, main = paste("Histogram for ",list[i]), xlab = "Smoking Habit")
}
```


Histogram for COPD
Frequency
0
5
10
15
20
25
30
1.0
1.5
2.0
2.5
3.0
Smoking Habit


Histogram for HC
Frequency
0
10
20
30
40
50
60
1.0
1.5
2.0
2.5
3.0
Smoking Habit


Histogram for Asthma
Frequency
0
10
20
30
40
50
60
70
1.0
1.5
2.0
2.5
3.0
Smoking Habit




```
# from dataset it is clear that numeric values are

#1:Non-Smoker
#2:Ex-Smoker
#3:Active Smoker
```

The above four histograms each represent the frequency of smoking habits (non-smokers, ex-smokers, active smokers) in different health conditions: COPD, HC (Healthy Control), Asthma, and an unspecified infection. Ex-smokers dominate in the HC group, while ex-smokers are more prevalent in the COPD and Asthma groups. Active smokers have the lowest frequency across all conditions except for the infection, where they are as frequent as Ex-smokers.

Co-Relation Matrix

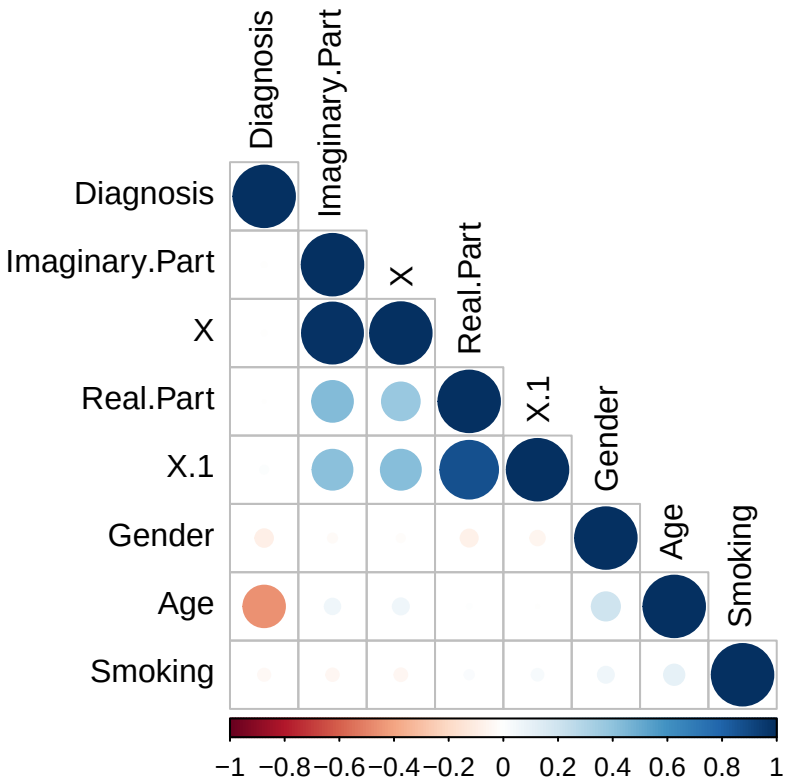
To find which variable has most impacted on target variable

```
#install.packages("corrplot")
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3
```

```
## corrplot 0.92 loaded
```

```
# Assuming your dataset is named 'dataset' and you want to exclude the column named 'exclude_column'
# Exclude the 'exclude_column' from the dataset
dataset_sub <- dataset[, !names(dataset) %in% c("ID")]
# Calculate the correlation matrix
correlation_matrix <- cor(dataset_sub)
corrplot(correlation_matrix, method = "circle", type = "lower", tl.col = "black")
```



#The above correlation matrix defines the size and color of the circles represent the strength and the direction of the correlation between different variables. Dark blue circles indicate a strong positive correlation, while dark red indicates a strong negative correlation, and lighter shades or smaller circles suggest weaker relationships. Variables like "Real.Part" and "X" as well as "Age" and "Smoking" show significant positive correlations, while there is no visible strong negative correlation between the variables presented.

Methods to predict the diagnosis

#The above dataset can be analysed using clustering and classification methods.clustering can be performed based on the non-label data where as the classification can be performed based on the label data

Classifications

#K-Nearest Neighbours

```
# Loading the required libraries
#install.packages('caret')
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

```
## Loading required package: lattice
```

```r
library(class)
# Splits the dataset into training(80%) and testing(20%) sets
# Loads the rpart package
library(rpart)
# Setting the seed for reproducibility
set.seed(123)
train_indices <- sample(1:nrow(dataset), 0.8*nrow(dataset))
train_data <- dataset[train_indices, ]
test_data <- dataset[-train_indices, ]
train_x <- train_data[, -1]
# All columns except the last one
test_x <- test_data[, -1]
train_y <- train_data[, 1]
test_y <- test_data[, 1]
# Applying KNN
k <- 5
# for optimization we choose k equal to 5, beacuse in most cases we can get optimization for k = 5
knn_pred <- knn(train_x, test_x, train_y, k)
# Calculating accuracy
accuracy <- sum(knn_pred == test_y) / length(test_y) * 100
cat("Accuracy of KNN model:", accuracy,"%\n")
```

```
## Accuracy of KNN model: 50 %
```

Confusion Matrix

```r
# Converts test_y to factor with the same levels as knn_pred
test_y <- factor(test_y, levels = levels(knn_pred))
# Creates confusion matrix
confusion_matrix <- confusionMatrix(data = factor(knn_pred, levels = levels(test_y)),reference = test_y
# Prints confusion matrix
print(confusion_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3  4
##          1  2  3  4  3
##          2  2 11  0  1
##          3  6  1 17  8
##          4  1  0  5  4
##
## Overall Statistics
##
##                Accuracy : 0.5
##                  95% CI : (0.3762, 0.6238)
##     No Information Rate : 0.3824
##     P-Value [Acc > NIR] : 0.0319
##
##                   Kappa : 0.2973
```

```
##
##  Mcnemar's Test P-Value : 0.6372
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4
## Sensitivity           0.18182   0.7333   0.6538  0.25000
## Specificity           0.82456   0.9434   0.6429  0.88462
## Pos Pred Value        0.16667   0.7857   0.5312  0.40000
## Neg Pred Value        0.83929   0.9259   0.7500  0.79310
## Prevalence            0.16176   0.2206   0.3824  0.23529
## Detection Rate        0.02941   0.1618   0.2500  0.05882
## Detection Prevalence  0.17647   0.2059   0.4706  0.14706
## Balanced Accuracy     0.50319   0.8384   0.6484  0.56731
```

The above confusion matrix is a table used to evaluate the performance of a classification model. The rows represent the model's predictions, while the columns represent the true classes or reference. The diagonal values (2 for class 1, 11 for class 2, 17 for class 3, and 4 for class 4) indicate correct predictions, while the off-diagonal numbers represent the instances where the model misclassified. For example, there are 6 instances where the model incorrectly predicted the true class 1 as class 3.

Decision Tree

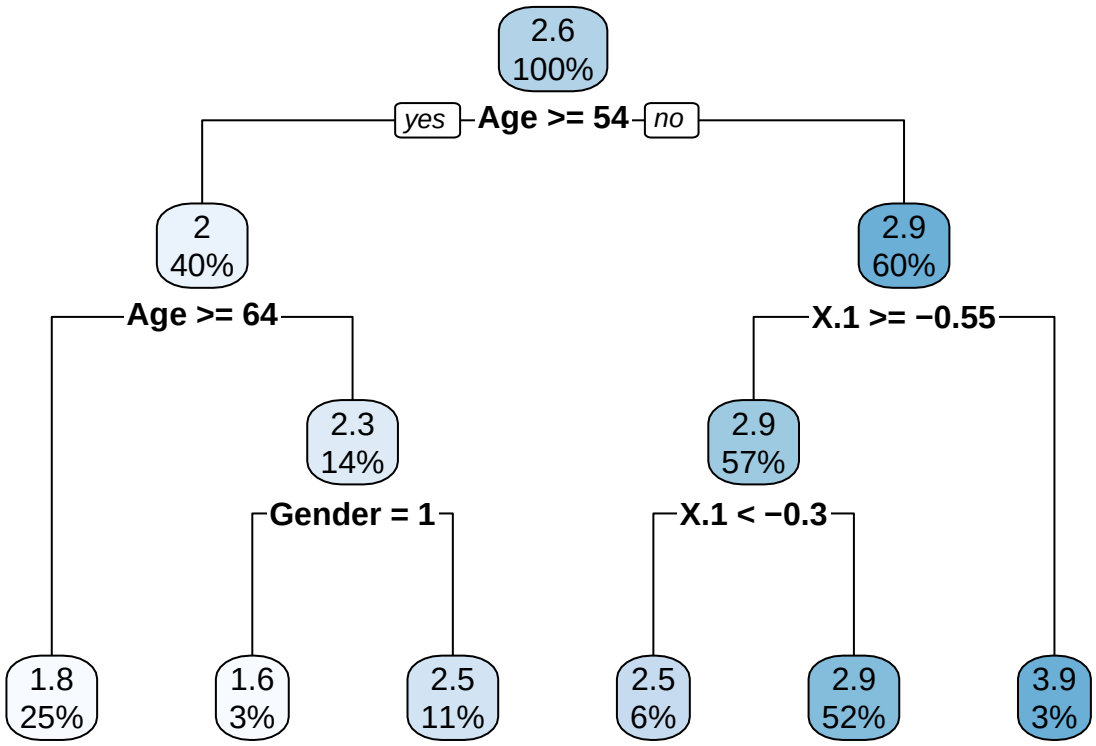
```r
# Load the rpart package
library(rpart)
# Sets the seed for reproducibility
set.seed(1)
# Splits the dataset into training (80%) and testing (30%) sets
train_indices <- sample(1:nrow(dataset), 0.8 * nrow(dataset))
train_data <- dataset[train_indices, ]
test_data <- dataset[-train_indices, ]
# Fits a decision tree model using the training data
tree_model <- rpart(Diagnosis ~ ., data = train_data, method = "class")
# Make predictions on the testing data
predictions <- predict(tree_model, test_data, type = "class")
# Calculates the accuracy
accuracy <- mean(predictions == test_data$Diagnosis)
cat("Accuracy:", accuracy*100,"%","\n")
```

```
## Accuracy: 58.82353 %
```

```r
#install.packages('rpart.plot')
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.3.3
```

```r
# Train the decision tree model
fit <- rpart(Diagnosis ~ ., data = train_data)
# Plot the decision tree
rpart.plot(fit)
```



The above decision tree starts with an initial node using age to split the data: individuals aged 54 and above go to the left, while those younger go to the right. For those 54 and older, a further split at age 64 differentiates the groups, with an additional split based on gender. On the right side, the tree splits based on the value of X.1, with further subdivision at X.1 = -0.3, leading to the terminal nodes with the final values ranging from 1.8 to 3.9, indicating the predicted outcomes or classes.

Confusion Marix

```r
# Confusion Matrix for Decision Tree
# Make predictions on the test data
tree_pred <- predict(tree_model, newdata = test_data, type = "class")
# Creates the confusion matrix
conf_matrix <- table(tree_pred, test_data$Diagnosis)
# Prints the confusion matrix
print(conf_matrix)
```

```
##
```

```
## tree_pred  1  2  3  4
##         1  3  3  7  0
##         2  2 14  0  0
##         3  5  0 22 10
##         4  0  0  1  1
```

#This confusion matrix shows the following: #- For class 1, 3 instances were predicted as class 1, 3 instances were predicted as class 2, 7 instances were predicted as class 3, and none were predicted as class 4. #- For class 2, 2 instances were predicted as class 1, 14 instances were predicted as class 2, and none were predicted as classes 3 and 4. #- For class 3, 5 instances were predicted as class 1, none were predicted as class 2, 22 instances were predicted as class 3, and 10 instances were predicted as class 4. #- For class 4, none were predicted as classes 1 and 2, 1 instance was predicted as class 3, and 1 instance was predicted as class 4.

Logistic Regression

```r
# Load the nnet library
#install.packages('nnet')
library(nnet)
# Trains the multinomial logistic regression model
logistic_model <- multinom(Diagnosis ~ ., data = train_data)
```

```
## # weights:  36 (24 variable)
## initial  value 374.299478
## iter  10 value 290.133765
## iter  20 value 269.376751
## iter  30 value 269.054851
## final  value 269.054745
## converged
```

```r
# Prints the details of the trained model
print(logistic_model)
```

```
## Call:
## multinom(formula = Diagnosis ~ ., data = train_data)
##
## Coefficients:
##   (Intercept) Imaginary.Part          X  Real.Part         X.1     Gender
## 2  -11.151235     -3.5142499  3.4335902  0.5797569 -0.01083497  1.1690161
## 3    3.404286      0.9612350 -0.5815009 -1.1945392  0.35364877  0.3464181
## 4    2.471917      0.7392752 -0.6095922 -0.5904049 -0.48018260 -0.4501603
##           Age    Smoking
## 2  0.12038058 1.50814509
## 3 -0.06771231 0.04962459
## 4 -0.06182037 0.26758461
##
## Residual Deviance: 538.1095
## AIC: 586.1095
```

```r
# Make predictions on new data
logistic_pred <- predict(logistic_model, newdata = test_data, type = "class")
# Evaluates the accuracy of the model
accuracy <- sum(logistic_pred == test_data$Diagnosis) / nrow(test_data)
cat("Accuracy for Logistic Regression Model:", accuracy * 100)
```

```
## Accuracy for Logistic Regression Model: 52.94118
```

```r
## Confusion Matrix for Decision Tree
# Make predictions on the test data
logistic_pred <- predict(logistic_model, newdata = test_data, type = "class")
# Creates confusion matrix
conf_matrix <- table(logistic_pred, test_data$Diagnosis)
# Prints the confusion matrix
print(conf_matrix)
```

```
##
## logistic_pred  1  2  3  4
##             1  4  3  6  2
##             2  2 13  4  0
##             3  2  1 19  9
##             4  2  0  1  0
```

#The above confusion matrix indicates:

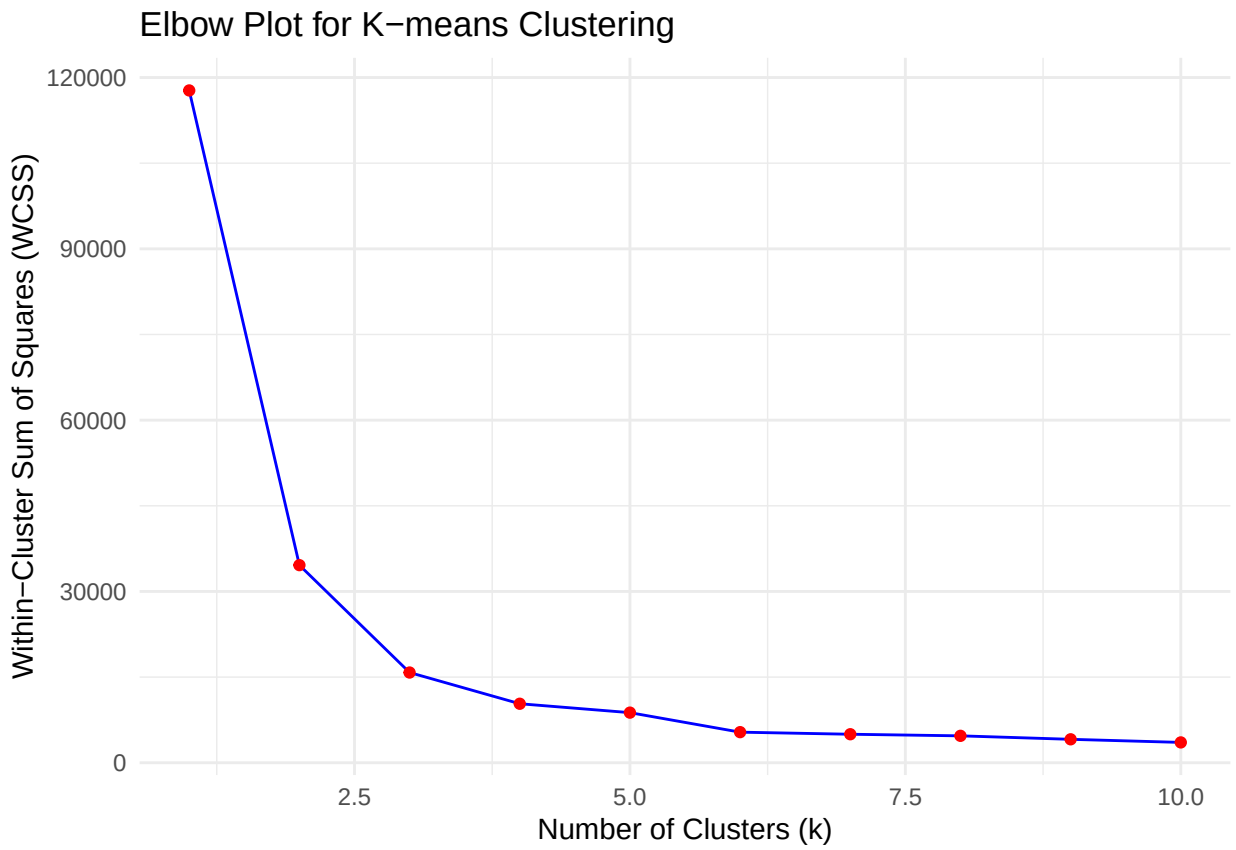
#- Class 1: Correctly predicted 4, misclassified as class 2: 3, misclassified as class 3: 6, misclassified as class 4: 2. #- Class 2: Misclassified as class 1: 2, correctly predicted 13, misclassified as class 3: 4, and no predictions for class 4. #- Class 3: Misclassified as class 1: 2, misclassified as class 2: 1, correctly predicted 19, misclassified as class 4: 9. #- Class 4: Misclassified as class 1: 2, no predictions for class 2, misclassified as class 3: 1, and no correct predictions for class 4.

#Clustering Methods # K-mean Clustering

```r
# Loads the required library
library(ggplot2)
#library(ggthemes)
# Runs the k-means clustering for a range of k values
sum_square <- numeric(length = 10)
# Initializes the vector to store sum_square values
for (i in 1:10) {
  kmeans_model <- kmeans(dataset, centers = i)
  sum_square[i] <- kmeans_model$tot.withinss  # Store WCSS value
}

#Plots the Within Cluster Sum of Squares values against the number of clusters (k)
elbow_plot <- ggplot(data = data.frame(k = 1:10, sumsquare = sum_square), aes(x = k, y = sum_square)) +
  geom_line(color = "blue") +
  geom_point(color = "red") +
  labs(title = "Elbow Plot for K-means Clustering",
       x = "Number of Clusters (k)",
       y = "Within-Cluster Sum of Squares (WCSS)") +
  theme_minimal()
```

```
#Displays the elbow plot
print(elbow_plot)
```

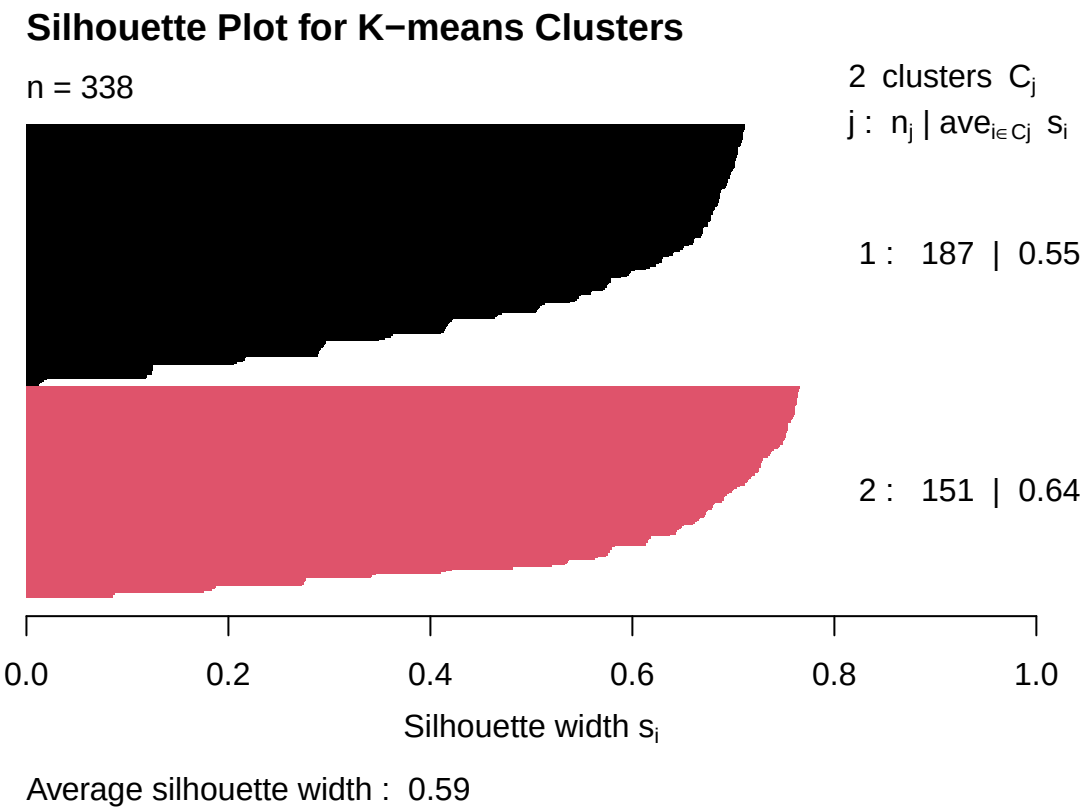


#If the elbow plot is to indicate k=2 as the optimal number of clusters, then the "elbow" or the point of inflection would be where the plot starts to flatten out after k=2, instead of k=3. This would mean that the reduction in within-cluster sum of squares (WCSS) from 1 to 2 is significant, while the reduction from 2 to 3 is not as pronounced, thus k=2 would be the most efficient choice for clustering this particular dataset. # for k = 2 we are getting the high silhouette value than k = 3

```
library(cluster)
#From above it is clear at 3 there can be optimal clustering
kmeans_result <- kmeans(dataset, centers = 2)
silhouette_widths <- silhouette(kmeans_result$cluster, dist(dataset))
# Average silhouette width
avg_silhouette_width <- mean(silhouette_widths[, "sil_width"])
print(paste("Silhouette score:", avg_silhouette_width))
```

```
## [1] "Silhouette score: 0.586235945928144"
```

```
# Plotting silhouette score
plot(silhouette_widths, col = 1:2, border = NA, main = "Silhouette Plot for K-means Clusters")
```



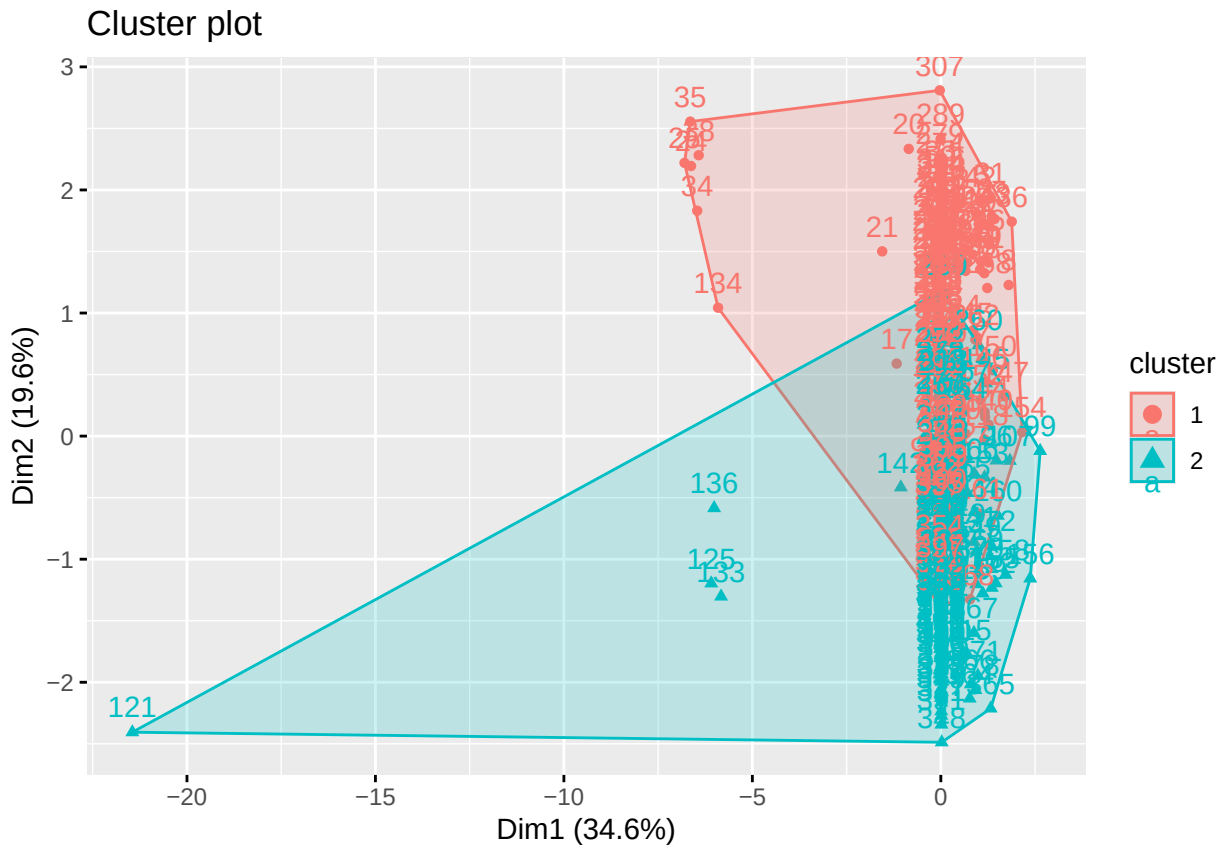
The above silhouette plot for K-means clustering with two clusters shows that Cluster 2 (156 data points) has a higher silhouette width (0.62), indicating a better fit than Cluster 1 (182 data points, silhouette width 0.55). The overall average silhouette width is 0.58, signifying a moderately strong cluster structure within the data.

```
#install.packages('factoextra')
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 4.3.3
```

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

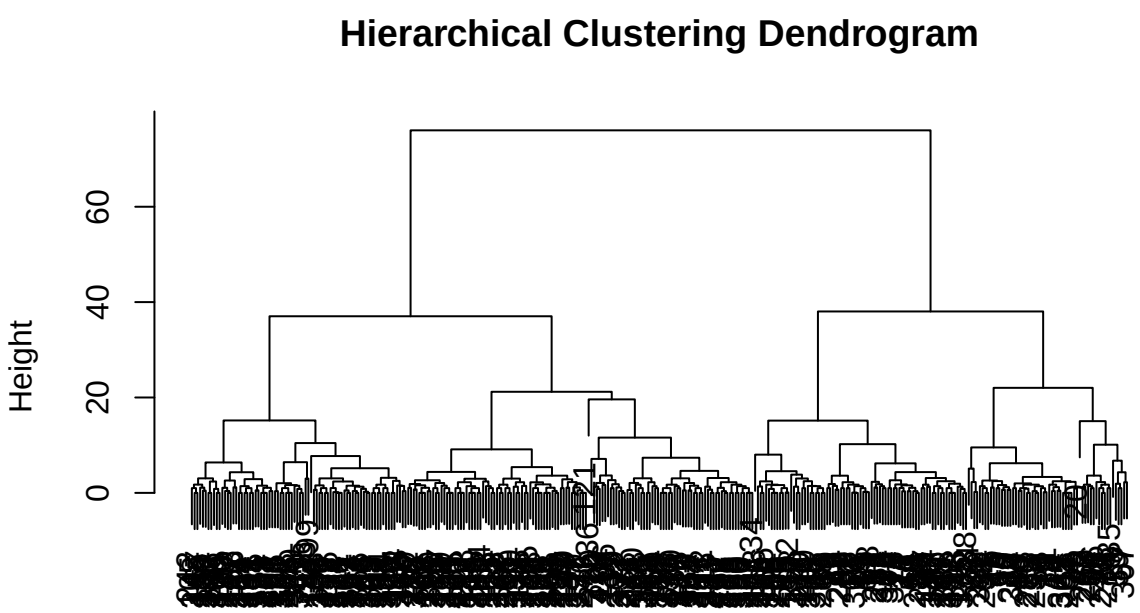
```
fviz_cluster(kmeans_result, data = dataset)
```



#The cluster plot depicts two clusters on a scatterplot, where Dim1 and Dim2 likely represent principal components accounting for 34.6% and 19.6% of the data's variance respectively. Cluster 1 (red circles) seems to be more dispersed than Cluster 2 (blue triangles), which is more densely packed and focused around the center of Dim2.

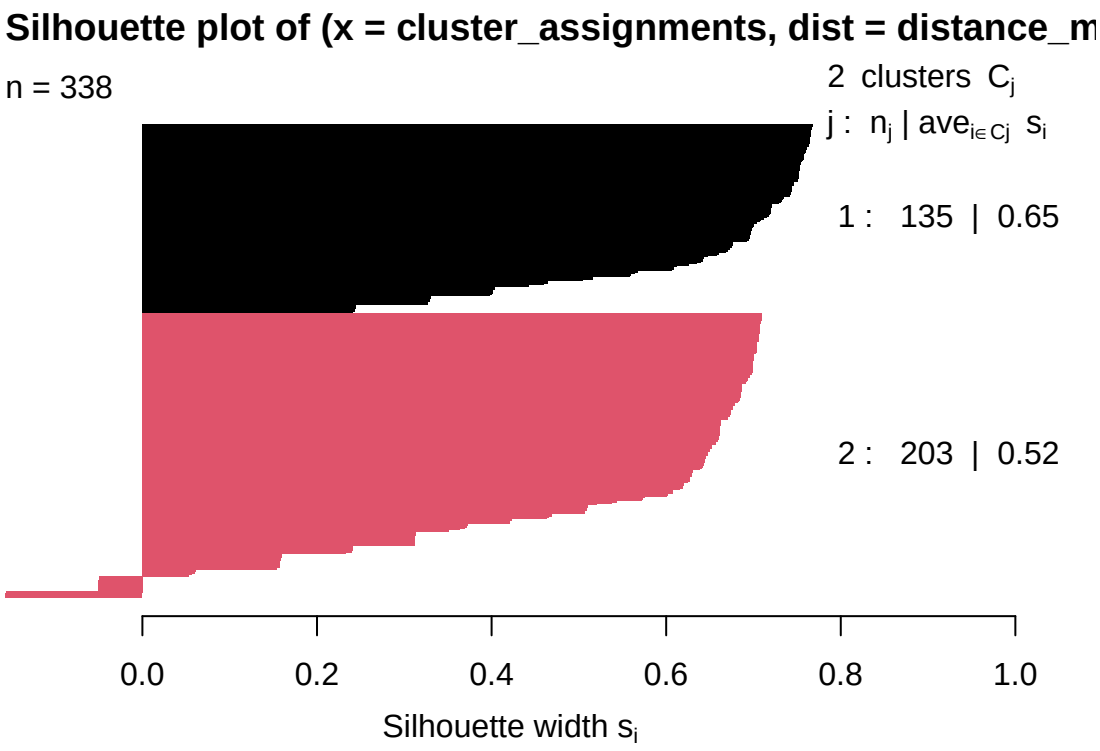
Hierarchical Clustering

```
distance_matrix <- dist(dataset[,c("Imaginary.Part","Real.Part",'Gender','Age','Smoking')])
# Performing hierarchical clustering using the complete linkage method
hc <- hclust(distance_matrix, method = "complete")
# Plotting the dendrogram
plot(hc, main = "Hierarchical Clustering Dendrogram",xlab="Data Points")
```



#The above dendrogram represents the results of hierarchical clustering using the complete linkage method, where each leaf represents a data point, and the height of the joins reflects the distance at which clusters are merged. The structure of the dendrogram indicates that certain clusters are very distinct and are joined only at the higher levels of the hierarchy, which could suggest the presence of distinct groups within the data.

```
#install.packages("cluster")
library(cluster)
# Cutting the tree to get cluster assignments for each point
k <- 2 # considering 5 clusters
cluster_assignments <- cutree(hc, k)
# Calculates the silhouette width for each observation
silhouette_widths <- silhouette(cluster_assignments, distance_matrix)
# Calculates the average silhouette score
avg_sil_width <- mean(silhouette_widths[, "sil_width"])
# Plot silhouette analysis
plot(silhouette_widths, col = 1:k, border = NA)
```



The silhouette plot depicts the clustering quality of 338 data points into two clusters, with silhouette coefficients ranging from -1 to 1. Cluster 1, with 135 points, has a silhouette width of 0.65 indicating a good cluster structure, whereas Cluster 2, with 203 points, has a slightly lower silhouette width of 0.52. The average silhouette width of 0.57 across both clusters suggests that the data points are, on average, reasonably well matched to their own clusters.

CONCLUSION

Finally, for clustering, K-means might be the preferred method for this dataset, while for classification purposes, the decision tree algorithm should be considered over KNN and logistic regression, given its superior performance in accuracy.