NETAJI SUBHAS UNIVERSITY OF TECHNOLOGY

COMPUTER HARDWARE SOFTWARE WORKSHOP

COCSC19

SUBMITTED BY:

KHUSHI KHARKE- 2021UCS1617

ADITYA YADAV- 2021UCS1640

YASH CHAUHAN- 2021UCS1628

R PROGRAMMING TASK

1) Explain Basic Data Structure in R.

In R, a programming language and environment designed for statistical computing and data analysis, there are several basic data structures that are commonly used. These data structures include vectors, matrices, arrays, lists, and data frames. Let's briefly explain each of them:

Vectors:

- A vector is the most basic data structure in R.
- It is a one-dimensional array that can hold numeric, character, or logical data.
- Elements in a vector must be of the same data type.
- You can create a vector using the c() function.

Matrices:

- A matrix is a two-dimensional array that can hold elements of the same data type.
- It is created using the matrix() function.
- Elements are arranged in rows and columns.

Arrays:

- An array is a multi-dimensional extension of a matrix.
- It can have more than two dimensions.
- You can create an array using the array() function.

Lists:

- A list is a versatile data structure that can hold elements of different data types.
- Elements can be vectors, matrices, arrays, or even other lists.
- Lists are created using the list() function.

Data Frames:

- A data frame is a two-dimensional table-like structure.
- It is similar to a matrix, but columns can have different data types.
- It is often used to store datasets.
- Data frames can be created using the data.frame() function.

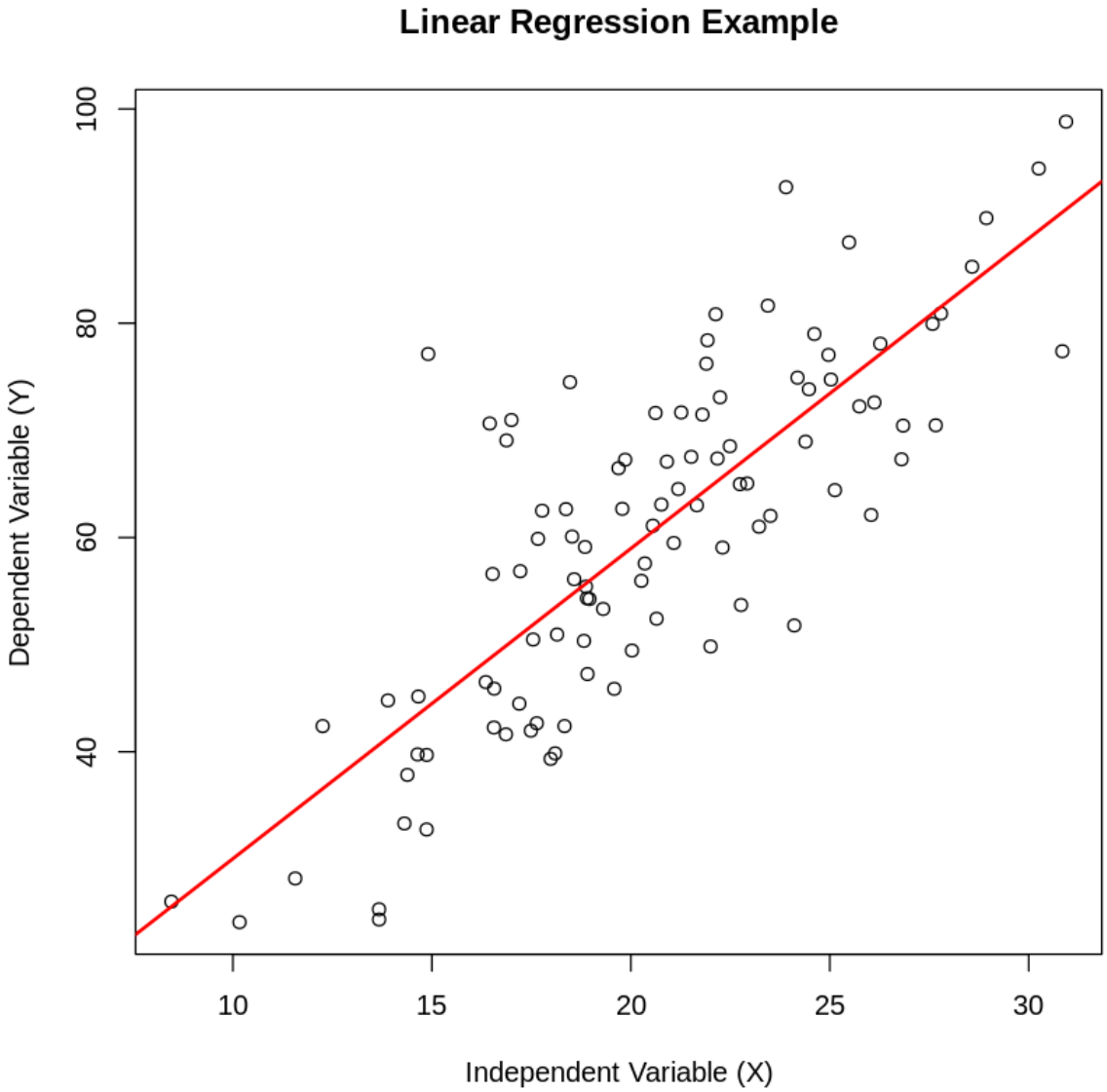
2) Implement Linear Regression in R and Visualize the results.

CODE:

```r
# Install and load the tidyverse package
install.packages("tidyverse")
library(tidyverse)

# Generate some example data
set.seed(123)
x <- rnorm(100, mean = 20, sd = 5)
y <- 3 * x + rnorm(100, mean = 0, sd = 10)

# Create a data frame with the generated data
data <- data.frame(x = x, y = y)

# Fit linear regression model
lm_model <- lm(y ~ x, data = data)

# Print the summary of the model
summary(lm_model)

# Visualize the results
plot(x, y, main = "Linear Regression Example", xlab = "Independent Variable (X)", ylab = "Dependent Variable (Y)")
abline(lm_model, col = "red", lwd = 2)
```

```
Call:
lm(formula = y ~ x, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-19.073  -6.835  -0.875   5.806  32.904

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.0708     4.4782   0.239    0.812
x             2.8951     0.2138  13.544   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.707 on 98 degrees of freedom
Multiple R-squared:  0.6518,    Adjusted R-squared:  0.6482
F-statistic: 183.4 on 1 and 98 DF,  p-value: < 2.2e-16
```

OUTPUT:


Linear Regression Example
Dependent Variable (Y)
Independent Variable (X)
100
80
60
40
10
15
20
25
30

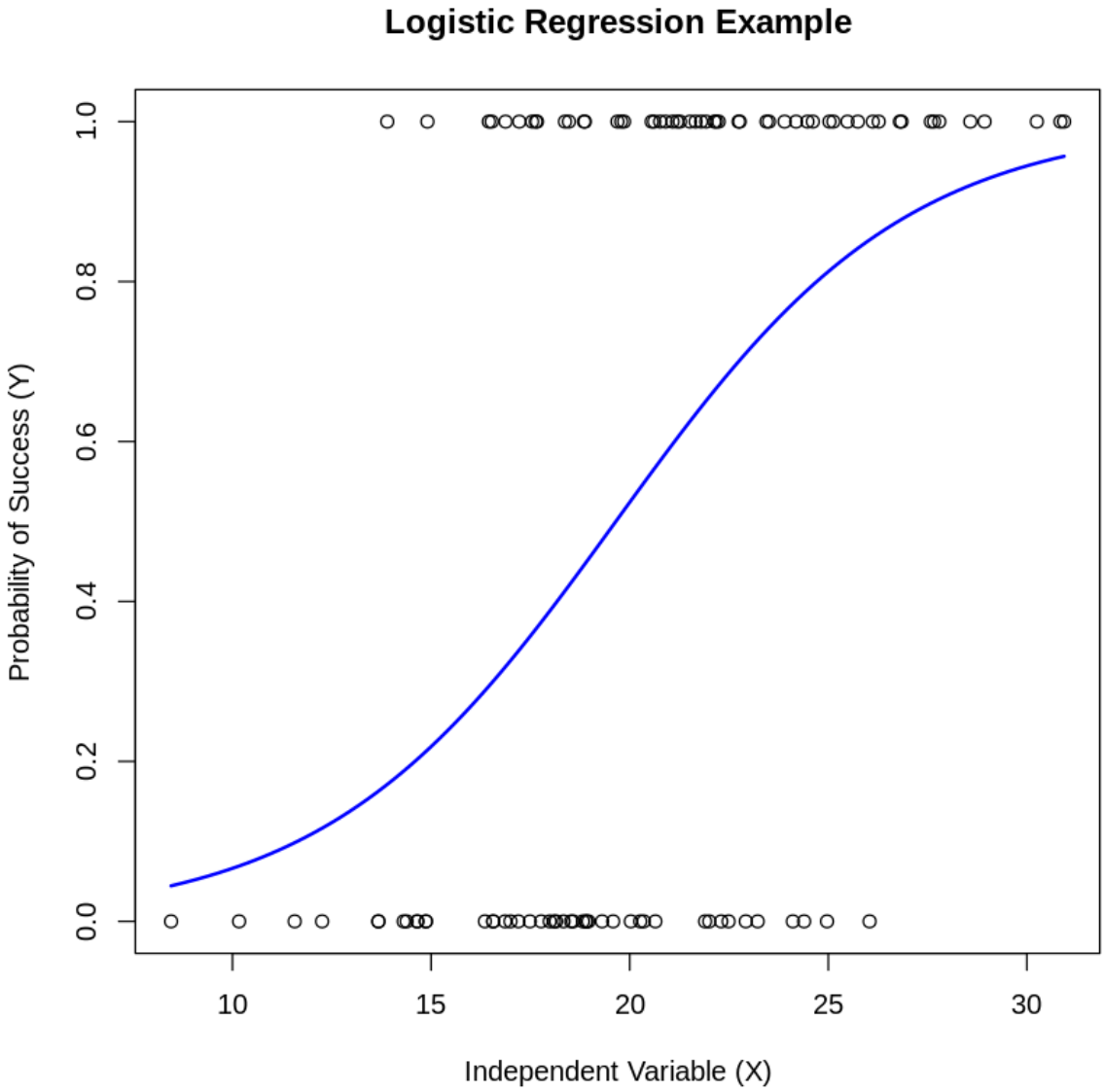
3) Implement Logistic Regression in R and Visualize the results.

CODE:

```r
# Install and load the tidyverse package
install.packages("tidyverse")
library(tidyverse)
# Generate some example data
set.seed(123)
x <- rnorm(100, mean = 20, sd = 5)
log_odds <- 0.5 * x - 10 + rnorm(100, mean = 0, sd = 2)
probabilities <- 1 / (1 + exp(-log_odds))
y <- ifelse(runif(100) < probabilities, 1, 0)

# Create a data frame with the generated data
data <- data.frame(x = x, y = y)

# Fit logistic regression model
logit_model <- glm(y ~ x, data = data, family = "binomial")

# Print the summary of the model
summary(logit_model)

# Visualize the results
plot(x, y, main = "Logistic Regression Example", xlab = "Independent Variable (X)", ylab = "Probability of Success (Y)")
curve(predict(logit_model, data.frame(x = x), type = "response"), col = "blue", lwd = 2, add = TRUE)
```

```
Call:
glm(formula = y ~ x, family = "binomial", data = data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -5.38580    1.32037  -4.079 4.52e-05 ***
x            0.27408    0.06486   4.226 2.38e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 137.99  on 99  degrees of freedom
Residual deviance: 112.29  on 98  degrees of freedom
AIC: 116.29

Number of Fisher Scoring iterations: 4
```

OUTPUT:


Logistic Regression Example
Probability of Success (Y)
0.0
0.2
0.4
0.6
0.8
1.0
10
15
20
25
30
Independent Variable (X)

4) Implement any Machine learning Algorithm along with feature selection and data visualization on any dataset of your choice.

CODE:

```r
# Load necessary libraries
library(randomForest)
library(caret)
library(ggplot2)

# Load your dataset
# Replace 'your_dataset.csv' with the actual file name and path
data <- read.csv("C:/Users/yashc/OneDrive/Desktop/arduino project/city_day.csv")

# Check the structure of the dataset
str(data)

# Explore summary statistics of the dataset
summary(data)

# Data preprocessing: Handle missing values, if any
data <- na.omit(data)

# Data visualization: Pair plot for exploratory analysis
pairs(data[, c("PM2.5", "PM10", "NO", "NO2", "NOx", "NH3", "SO2", "O3", "Benzene", "Toluene", "Xylene")])

# Feature selection: Using correlation matrix to identify highly correlated features
cor_matrix <- cor(data[, c("PM2.5", "PM10", "NO", "NO2", "NOx", "NH3", "SO2", "O3", "Benzene", "Toluene", "Xylene")])
highly_correlated_features <- findCorrelation(cor_matrix, cutoff = 0.8)

# Remove highly correlated features
selected_features <- colnames(data[, c("PM2.5", "PM10", "NO", "NO2", "NOx", "NH3", "SO2", "O3", "Benzene", "Toluene", "Xylene")]
[-highly_correlated_features])
data_selected <- data[, c(selected_features, "AQI")]
```

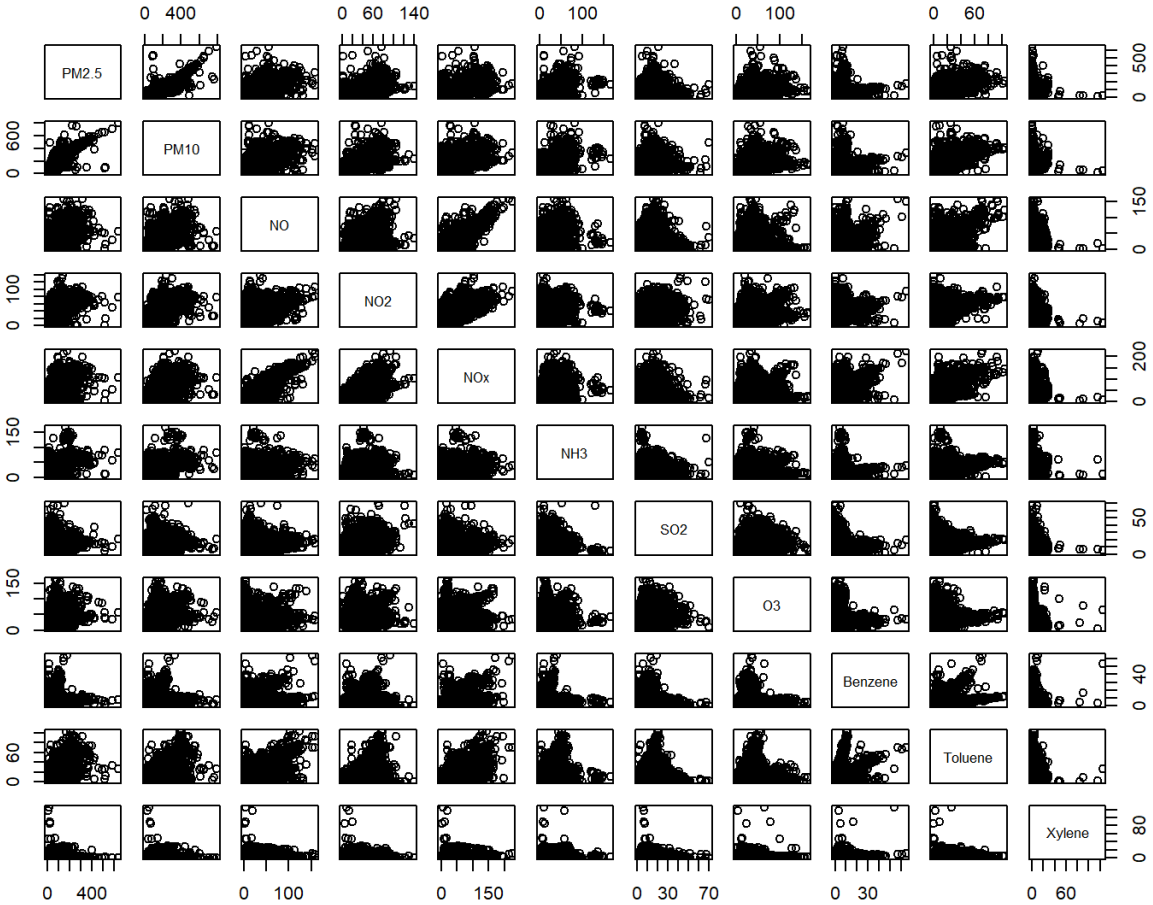
```r
# Split the dataset into training and testing sets
set.seed(123)  # for reproducibility
splitIndex <- createDataPartition(data_selected$AQI, p = 0.7, list = FALSE)
train_data <- data_selected[splitIndex, ]
test_data <- data_selected[-splitIndex, ]

# Train Random Forest model
rf_model <- randomForest(AQI ~ ., data = train_data, ntree = 100)

# Make predictions on the test set
predictions <- predict(rf_model, newdata = test_data)

# Evaluate the model
confusionMatrix(predictions, test_data$AQI)

# Feature importance plot
varImpPlot(rf_model)

# Visualize actual vs. predicted values
ggplot(data = data.frame(Actual = test_data$AQI, Predicted = predictions)) +
  geom_point(aes(x = Actual, y = Predicted)) +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Actual vs. Predicted AQI Values", x = "Actual AQI", y = "Predicted AQI")
```

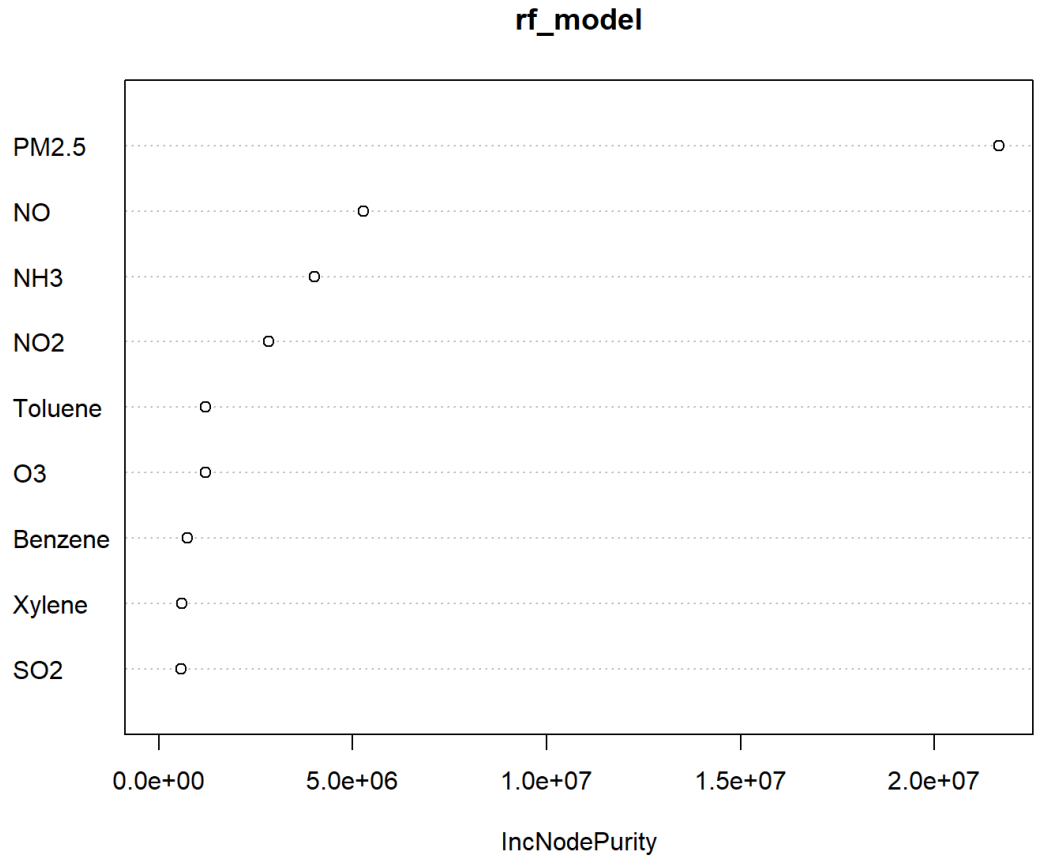
OUTPUTS:

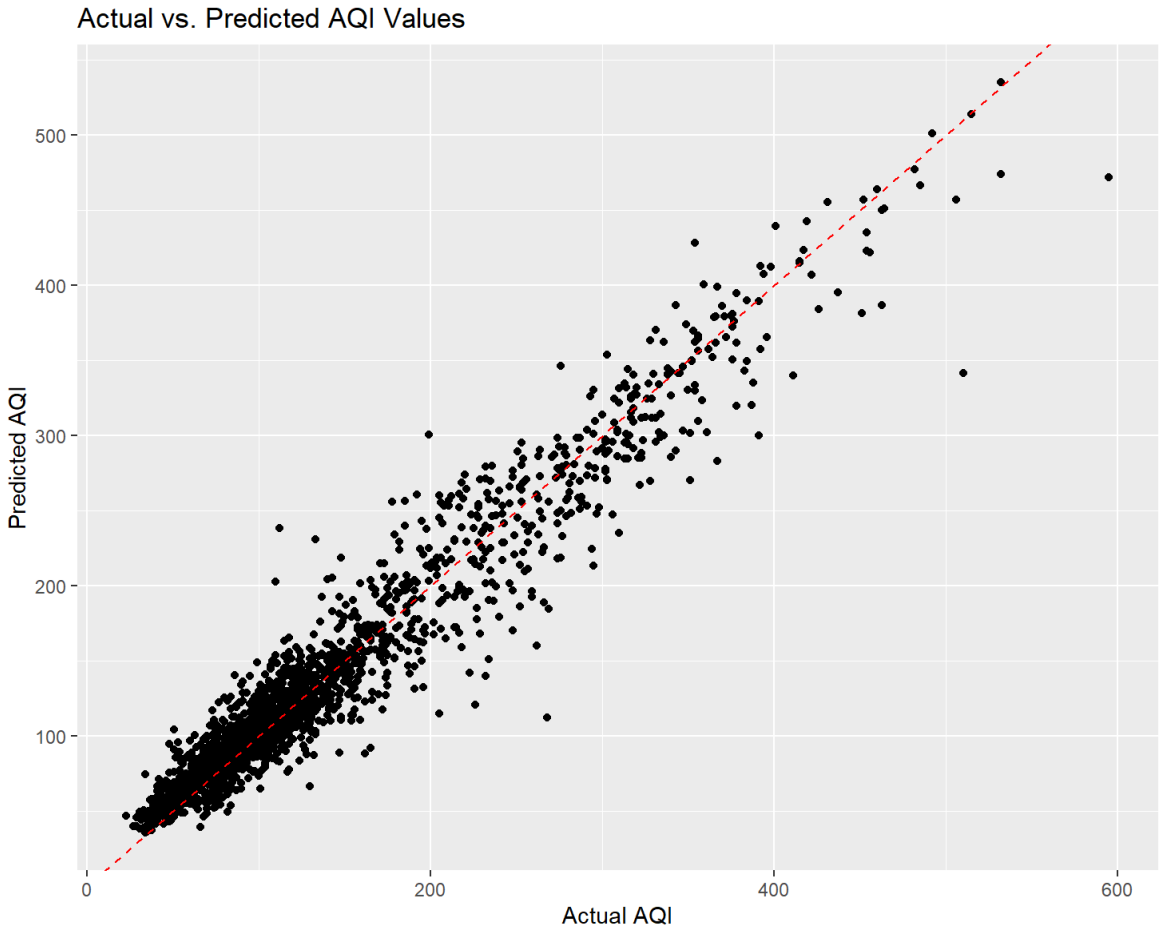
```
> str(data)
'data.frame':   29531 obs. of  16 variables:
 $ City      : chr  "Ahmedabad" "Ahmedabad" "Ahmedabad" "Ahmedabad" ...
 $ Date      : chr  "2015-01-01" "2015-01-02" "2015-01-03" "2015-01-04" ...
 $ PM2.5     : num  NA NA NA NA NA NA NA NA NA NA ...
 $ PM10      : num  NA NA NA NA NA NA NA NA NA NA ...
 $ NO        : num  0.92 0.97 17.4 1.7 22.1 ...
 $ NO2       : num  18.2 15.7 19.3 18.5 21.4 ...
 $ NOx       : num  17.1 16.5 29.7 18 37.8 ...
 $ NH3       : num  NA NA NA NA NA NA NA NA NA NA ...
 $ CO        : num  0.92 0.97 17.4 1.7 22.1 ...
 $ SO2       : num  27.6 24.6 29.1 18.6 39.3 ...
 $ O3        : num  133.4 34.1 30.7 36.1 39.3 ...
 $ Benzene   : num  0 3.68 6.8 4.43 7.01 5.42 0 0 0 0 ...
 $ Toluene   : num  0.02 5.5 16.4 10.14 18.89 ...
 $ Xylene    : num  0 3.77 2.25 1 2.78 1.93 0 0 0 0 ...
 $ AQI       : num  NA NA NA NA NA NA NA NA NA NA ...
 $ AQI_Bucket: chr  "" "" "" "" ...
```

```
> summary(data)
     City                Date               PM2.5              PM10
 Length:29531       Length:29531       Min.   :  0.04   Min.   :   0.01
 Class :character   Class :character   1st Qu.: 28.82   1st Qu.:  56.26
 Mode  :character   Mode  :character   Median : 48.57   Median :  95.68
                                       Mean   : 67.45   Mean   : 118.13
                                       3rd Qu.: 80.59   3rd Qu.: 149.75
                                       Max.   :949.99   Max.   :1000.00
                                       NA's   :4598     NA's   :11140
       NO                NO2                NOx                NH3
 Min.   :  0.02   Min.   :  0.01   Min.   :  0.00   Min.   :  0.01
 1st Qu.:  5.63   1st Qu.: 11.75   1st Qu.: 12.82   1st Qu.:  8.58
 Median :  9.89   Median : 21.69   Median : 23.52   Median : 15.85
 Mean   : 17.57   Mean   : 28.56   Mean   : 32.31   Mean   : 23.48
 3rd Qu.: 19.95   3rd Qu.: 37.62   3rd Qu.: 40.13   3rd Qu.: 30.02
 Max.   :390.68   Max.   :362.21   Max.   :467.63   Max.   :352.89
 NA's   :3582     NA's   :3585     NA's   :4185     NA's   :10328
       CO                SO2                O3              Benzene
 Min.   :  0.000   Min.   :  0.01   Min.   :  0.01   Min.   :  0.000
 1st Qu.:  0.510   1st Qu.:  5.67   1st Qu.: 18.86   1st Qu.:  0.120
 Median :  0.890   Median :  9.16   Median : 30.84   Median :  1.070
 Mean   :  2.249   Mean   : 14.53   Mean   : 34.49   Mean   :  3.281
 3rd Qu.:  1.450   3rd Qu.: 15.22   3rd Qu.: 45.57   3rd Qu.:  3.080
 Max.   :175.810   Max.   :193.86   Max.   :257.73   Max.   :455.030
 NA's   :2059      NA's   :3854     NA's   :4022     NA's   :5623
    Toluene            Xylene             AQI          AQI_Bucket
 Min.   :  0.000   Min.   :  0.00   Min.   :  13.0   Length:29531
 1st Qu.:  0.600   1st Qu.:  0.14   1st Qu.:  81.0   Class :character
 Median :  2.970   Median :  0.98   Median : 118.0   Mode  :character
 Mean   :  8.701   Mean   :  3.07   Mean   : 166.5
 3rd Qu.:  9.150   3rd Qu.:  3.35   3rd Qu.: 208.0
 Max.   :454.850   Max.   :170.37   Max.   :2049.0
 NA's   :8041      NA's   :18109    NA's   :4681
```

Data	
cor_matrix	num [1:11, 1:11] 1 0.896 0.604 0.559 0.619 ...
data	6236 obs. of 16 variables
data_selected	6236 obs. of 10 variables
rf_model	List of 18
splitIndex	int [1:4367, 1] 1 3 5 6 10 11 14 15 16 17 ...
test_data	1869 obs. of 10 variables
train_data	4367 obs. of 10 variables
Values	
highly_correlated…	int [1:2] 2 5
predictions	Named num [1:1869] 168 141 165 204 185 ...
selected_features	chr [1:9] "PM2.5" "NO" "NO2" "NH3" "SO2" "O3" "Be…




rf_model
PM2.5
NO
NH3
NO2
Toluene
O3
Benzene
Xylene
SO2
0.0e+00
5.0e+06
1.0e+07
1.5e+07
2.0e+07
IncNodePurity


Actual vs. Predicted AQI Values
Predicted AQI
500
400
300
200
100
0
200
400
600
Actual AQI