CAPSTONE PROJECT - 2

Computer Hardware Software Workshop
COCSC19

Submitted by:-
Kanishk Kumar (2021UCS1627)
Shubham Yadav (2021UCS1630)
Kashish Goel (2021UCS1633)
CSE Section-2

Objective:-

Utilizing R library, develop a project on data visualization and analytics and perform the following tasks:-

a. Explain Basic Data Structure in R.
b. Implement Linear Regression in R and Visualize the results.
c. Implement Logistic Regression in R and Visualize the results.
d. Implement any Machine learning Algorithm along with feature selection and data visualization on any dataset of your choice.

R (Programming Language):-

R is a powerful and open-source programming language and software environment designed for statistical computing, data analysis, and graphics. Developed by statisticians and data scientists, R provides a comprehensive set of tools for manipulating, analyzing, and visualizing data. Known for its extensive collection of packages and libraries, R enables users to perform a wide range of statistical analyses, machine learning tasks, and data visualization techniques. With a syntax that is intuitive for statisticians and analysts, R has gained popularity in academia, research, and industries for its versatility and flexibility in handling diverse data types and structures. The R community actively contributes to the development of packages, making it a dynamic and evolving language that continues to play a central role in the field of data science and statistical computing.

A)Basic Data Structures in R:-

Vectors:

A vector is an ordered collection of basic data types of a given length. The only key thing here is all the elements of a vector must be of the identical data type e.g homogeneous data structures. Vectors are one-dimensional data structures.

eg :

X = c(1, 3, 5, 7, 8)

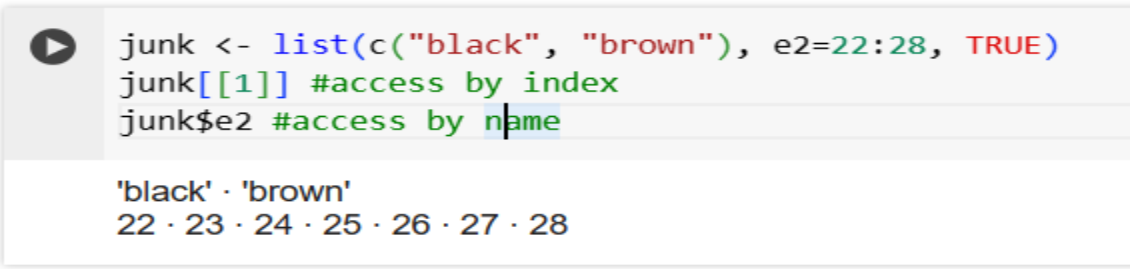
```
[1]  X = c(1, 3, 5, 7, 8)
     X[3]

     5
```

Lists:

A list is a generic object consisting of an ordered collection of objects. Lists are heterogeneous data structures. These are also one-dimensional data structures. A list can be a list of vectors, list of matrices, a list of characters and a list of functions and so on.

eg:

junk <- list(c("black", "brown"), e2=22:28, TRUE)



Dataframes:

Dataframes are generic data objects of R which are used to store the tabular data. Dataframes are the foremost popular data objects in R programming because we are comfortable in seeing the data within the tabular form. They are two-dimensional, heterogeneous data structures. These are lists of vectors of equal lengths.

Data frames have the following constraints placed upon them:

A data-frame must have column names and every row should have a unique name.

Each column must have the identical number of items.

Each item in a single column must be of the same data type.

Different columns may have different data types

eg:

df <- data.frame(haircolor = c("red", "black", "blonde"), eyecolor =c("green", "brown", "blue"), age = c(22, 23, 25), phd = c(T, T, F))

```
[ ] df <- data.frame(haircolor = c("red", "black", "blonde"), eyecolor =c("green", "brown", "blue"), age = c(22, 23, 25), phd = c(T, T, F))
    df
    df$age
```

A data.frame: 3 × 4

haircolor	eyecolor	age	phd
<chr>	<chr>	<dbl>	<lgl>
red	green	22	TRUE
black	brown	23	TRUE
blonde	blue	25	FALSE

22 · 23 · 25

Matrices:

A matrix is a rectangular arrangement of numbers in rows and columns. In a matrix, as we know rows are the ones that run horizontally and columns are the ones that run vertically. Matrices are two-dimensional, homogeneous data structures.

eg:

A = matrix(c(1, 2, 3, 4, 5, 6, 7, 8, 9), nrow = 3, ncol = 3,byrow=TRUE)

```
[ ] mat <- matrix(NA, nrow = 4, ncol = 5)
    mat[1,] <- c(2,3,7,5,8)
    mat[2,] <- rep(4, 5)
    mat[3,] <- 33:37
    mat[4,] <- seq(23, 31, 2)
    mat
```

A matrix: 4 × 5 of type dbl

2	3	7	5	8
4	4	4	4	4
33	34	35	36	37
23	25	27	29	31

Arrays:

Arrays are the R data objects which store the data in more than two dimensions. Arrays are n-dimensional data structures.

eg:

my_arr <- array(1:8, c(2,2,2))

```
[ ]  my_arr <- array(1:8, c(2,2,2))
     my_arr[,,1]
```

A matrix: 2 × 2 of type int

1	3
2	4

Dataset Summary:-

We have used the dataset which comprises of precipitation at different average temperature, humidity, and surface level pressure and Decision denotes that whether it will rain at that temperature, humidity and Pressure. Below is a sample of the few observations from the dataset.

temperature	humidity	precipitation	Pressure	Decision
<dbl>	<int>	<dbl>	<dbl>	<int>
25.8	91	0.1	977.1	0
24.3	96	0.9	977.9	1
25.3	93	0.4	978.1	0
27.8	83	0.1	978.5	0
28.9	80	0.0	978.8	0
29.8	77	0.2	978.7	0
30.7	73	0.1	978.6	0
31.4	70	0.1	978.0	0
32.0	68	0.1	976.8	0
32.9	64	0.1	975.7	0
32.8	62	0.0	974.5	0

26.2	94	7.3	977.4	1
26.2	96	1.9	977.3	1
26.1	96	0.9	977.4	1
26.6	95	0.1	977.8	0
27.1	94	0.4	978.4	0
28.1	88	1.1	978.1	1
28.9	83	2.4	978.1	1
29.7	79	0.3	978.3	0
⋮	⋮	⋮	⋮	⋮
27.3	93	3.2	976.7	1
26.6	96	4.9	977.1	1
26.6	96	0.6	976.0	0
26.5	97	0.3	975.1	0
26.4	98	0.3	974.4	0
26.5	97	1.1	974.8	1
28.1	87	1.2	974.7	1
28.2	83	0.2	975.0	0

B)Linear regression in R:-

Theory:
It is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables. One variable denoted x is regarded as an independent variable and the other one denoted y is regarded as a dependent variable. It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

- The dependent variable, also known as the response or outcome variable, is represented by the letter Y.
- The independent variable, often known as the predictor or explanatory variable, is denoted by the letter X.
- The intercept, or value of Y when X is zero, is represented by the β_0.
- The slope or change in Y resulting from a one-unit change in X is represented by the β_1.
- The error term or the unexplained variation in Y is represented by the ε.

The line which best fits is called the Regression line.
The equation of the regression line is given by:

$y = a + bx$

Where y is the predicted response value, a is the y-intercept, x is the feature value and b is the slope.

Implementing Linear Regression in R and Visualizing the results:

```
[21] data<-read.csv("/content/rainfall_pred.csv", fileEncoding = "Latin1", check.names = F)
```

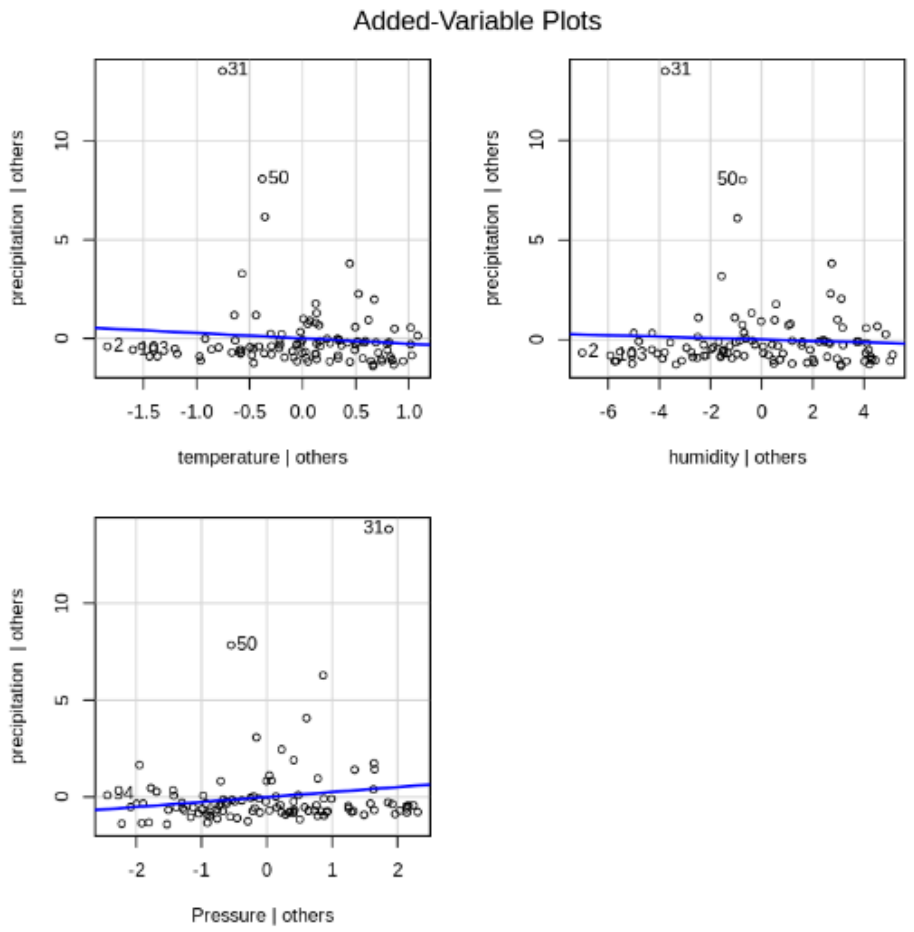
```
[22] names(data)
```

'temperature' · 'humidity' · 'precipitation' · 'Pressure' · 'Decision'

```
[23] model<-lm(precipitation~temperature+humidity+Pressure,data=data)
     summary(model)
```

```
Call:
lm(formula = precipitation ~ temperature + humidity + Pressure,
    data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-1.3852 -0.8825 -0.4952  0.0332 13.3392

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -238.99202  156.67585  -1.525    0.130
temperature   -0.27216    0.28367  -0.959    0.340
humidity      -0.03588    0.06184  -0.580    0.563
Pressure       0.25654    0.15944   1.609    0.111

Residual standard error: 1.918 on 102 degrees of freedom
Multiple R-squared:  0.04162,   Adjusted R-squared:  0.01343
F-statistic: 1.477 on 3 and 102 DF,  p-value: 0.2254
```

```r
#load car package
library(car)

#produce added variable plots
avPlots(model)
```



The following graphs show added variable plots of the dataset ,that is,plots the dependent variable plotted against each of the independent variables.

C)Logistic Regression in R:-

Theory:

Logistics regression is also known as a generalized linear model. As it is used as a classification technique to predict a qualitative response, the Value of y ranges from 0 to 1 and can be represented by the following equation:

$$\text{Odds} = \frac{P}{1 - P}$$

p is the probability of characteristic of interest. The odds ratio is defined as the probability of success in comparison to the probability of failure. It is a key representation of logistic regression coefficients and can take values between 0 and infinity. The odds ratio of 1 is when the probability of success is equal to the probability of failure. The odds ratio of 2 is when the probability of success is twice the probability of failure. The odds ratio of 0.5 is when the probability of failure is twice the probability of success.

$$\log(\text{Odds}) = \log\left(\frac{P}{1 - P}\right)$$

Since we are working with a binomial distribution(dependent variable), we need to choose a link function that is best suited for this distribution.

$$\text{logit}(P) = \log\left(\frac{P}{1 - P}\right) = b_0 + b_1*1 + b_2*2 + b_3*3 + \ldots + b_k*k$$

It is a logit function. In the equation above, the parenthesis is chosen to maximize the likelihood of observing the sample values rather than minimizing the sum of squared errors(like ordinary regression). The logit is also known as a log of odds. The logit function must be linearly related to the independent variables. This is from equation A, where the left-hand side is a linear combination of x. This is similar to the OLS assumption that y be linearly related to x. Variables b0, b1, b2 … etc are unknown and must be estimated on available training data. In a logistic regression model, multiplying b1 by one unit changes the logit by b0. The P changes due to a one-unit change will depend upon the value multiplied. If b1 is positive then P will increase and if b1 is negative then P will decrease.

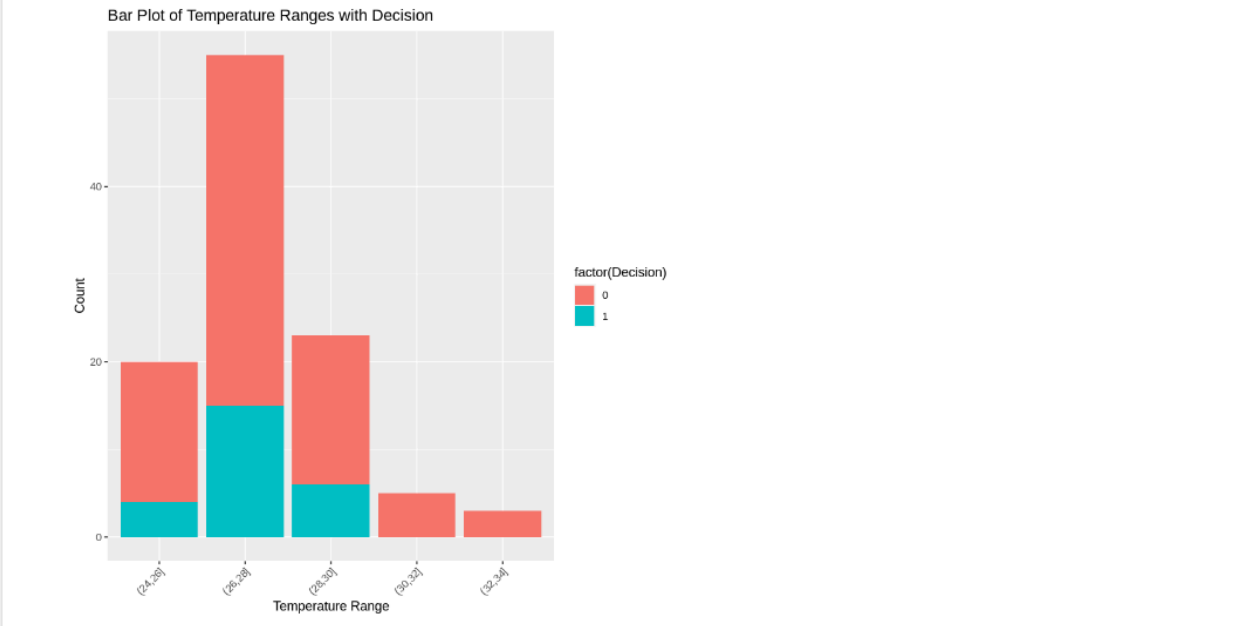
Implementing Logistic Regression in R and Visualizing the results:

```
[7]  library(ggplot2)
```

```
[6]  data1<-read.csv("/content/rainfall_pred.csv", fileEncoding = "Latin1", check.names = F)
```

```
[8]  data1$Decision = as.factor(data1$Decision)
```

```
temperature_ranges <- cut(data1$temperature, breaks = seq(20, 34, by = 2), include.lowest = TRUE)
# Create a new column 'Temperature Range' in the data frame
data1$Temperature_Range <- temperature_ranges
# Plot the bar chart with grouped temperature ranges
ggplot(data1, aes(Temperature_Range, fill = factor(Decision))) + geom_bar() +
  labs(title = "Bar Plot of Temperature Ranges with Decision", x = "Temperature Range", y = "Count")
  + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The graph shows the number of hours it rained or did not depending upon the range of temperatures

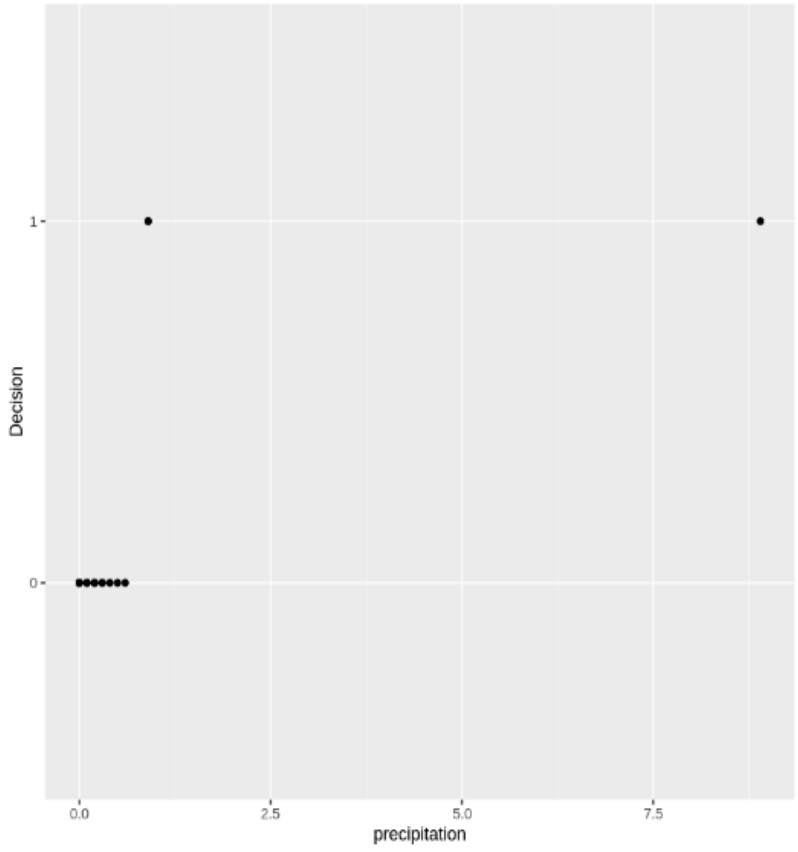
```
[25] library(caTools)
     library(ROCR)
```

```
[10] split <- sample.split(data1, SplitRatio = 0.8)

     train_reg <- subset(data1, split == "TRUE")
     test_reg <- subset(data1, split == "FALSE")

     # Training model
     logistic_model <- glm(Decision ~ temperature + humidity + Pressure, data = train_reg, family = "binomial")

     # Summarize the model
     summary(logistic_model)

     predict_reg <- predict(logistic_model,
                            test_reg, type = "response")
```

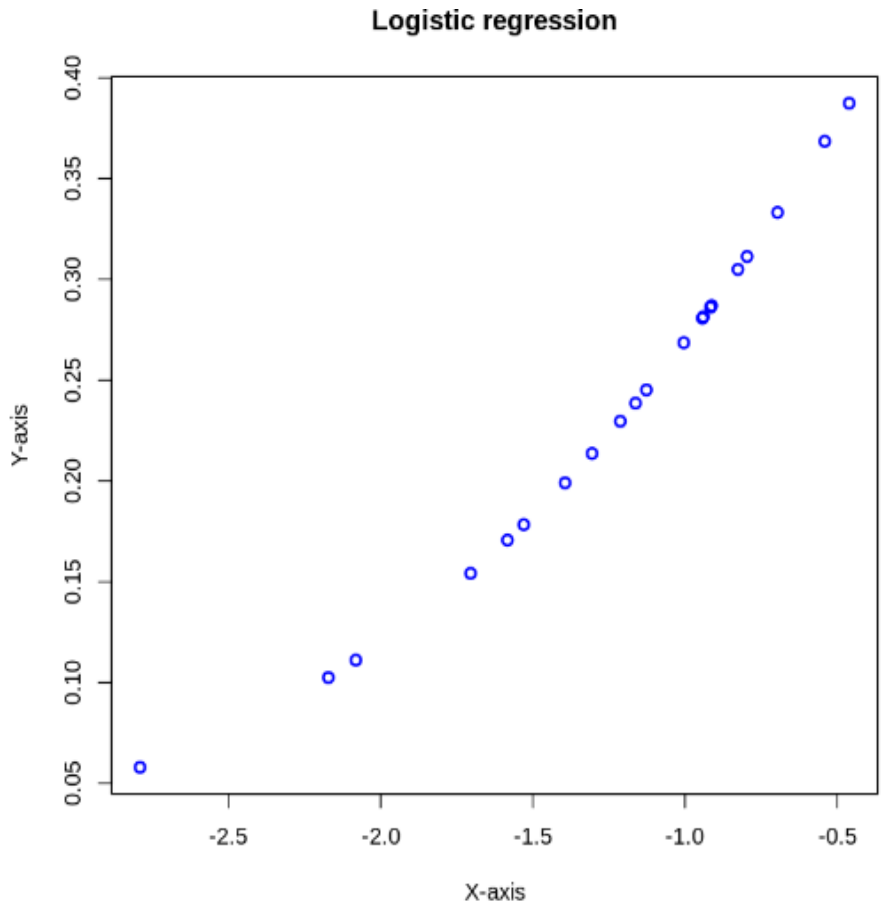
```
Call:
glm(formula = Decision ~ temperature + humidity + Pressure, family = "binomial",
    data = train_reg)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -288.75453  223.01647  -1.295    0.195
temperature   -0.01476    0.36350  -0.041    0.968
humidity       0.05174    0.08389   0.617    0.537
Pressure       0.29019    0.22604   1.284    0.199

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 97.210  on 84  degrees of freedom
Residual deviance: 93.407  on 81  degrees of freedom
AIC: 101.41

Number of Fisher Scoring iterations: 4
```

```
[13] #-0.01476*temperature+0.05174*humidity+0.29019*Pressure-288.75453
     ggplot(test_reg, aes(x=precipitation, y=Decision)) + geom_point() +
           stat_smooth(method="glm", color="green", se=FALSE,
                       method.args = list(family=binomial))
     plot(-0.01476*test_reg$temperature+0.05174*test_reg$humidity+0.29019*test_reg$Pressure-288.75453,
     predict_reg, col = "blue", lwd = 2, xlab = "X-axis",
          ylab = "Y-axis", main = "Logistic regression")
```

```
`geom_smooth()` using formula = 'y ~ x'
Warning message:
“Computation failed in `stat_smooth()`
Caused by error:
! y values must be 0 <= y <= 1”
```



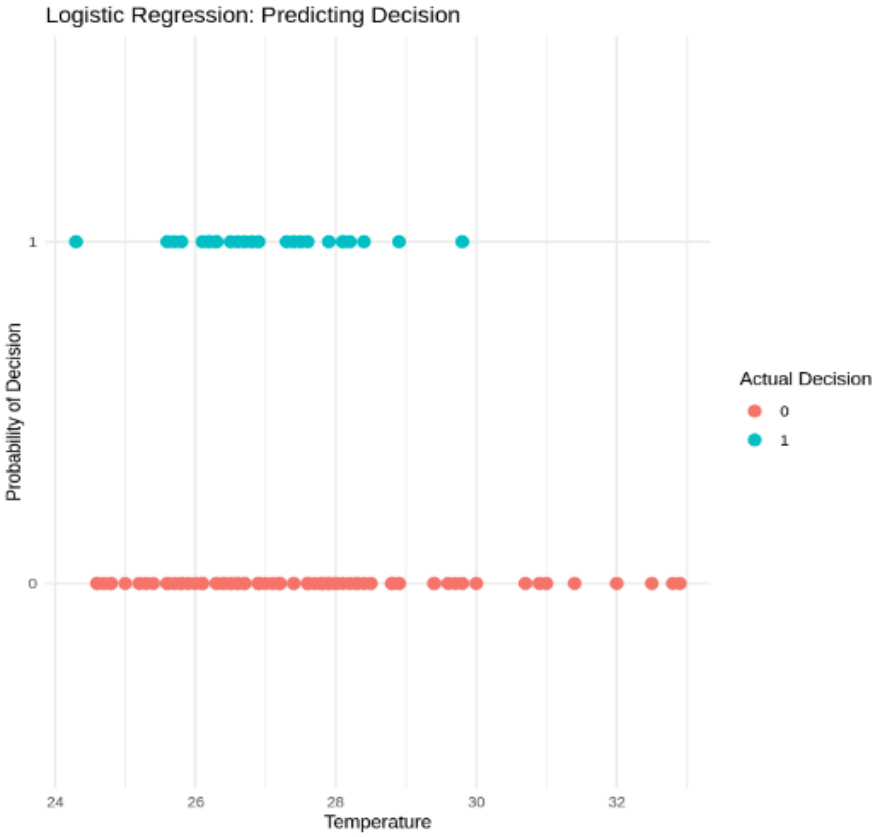
Logistic regression



The following curve displays the exponential logistic curve where X-axis id the linear equation while the Y-axis is the probability of raining

```
[14] # Visualize the results
     ggplot(data1, aes(x = temperature, y = Decision)) +
       geom_point(aes(color = factor(Decision)), size = 3) +
       geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
       labs(title = "Logistic Regression: Predicting Decision",
            x = "Temperature",
            y = "Probability of Decision",
            color = "Actual Decision") +
       theme_minimal()
```

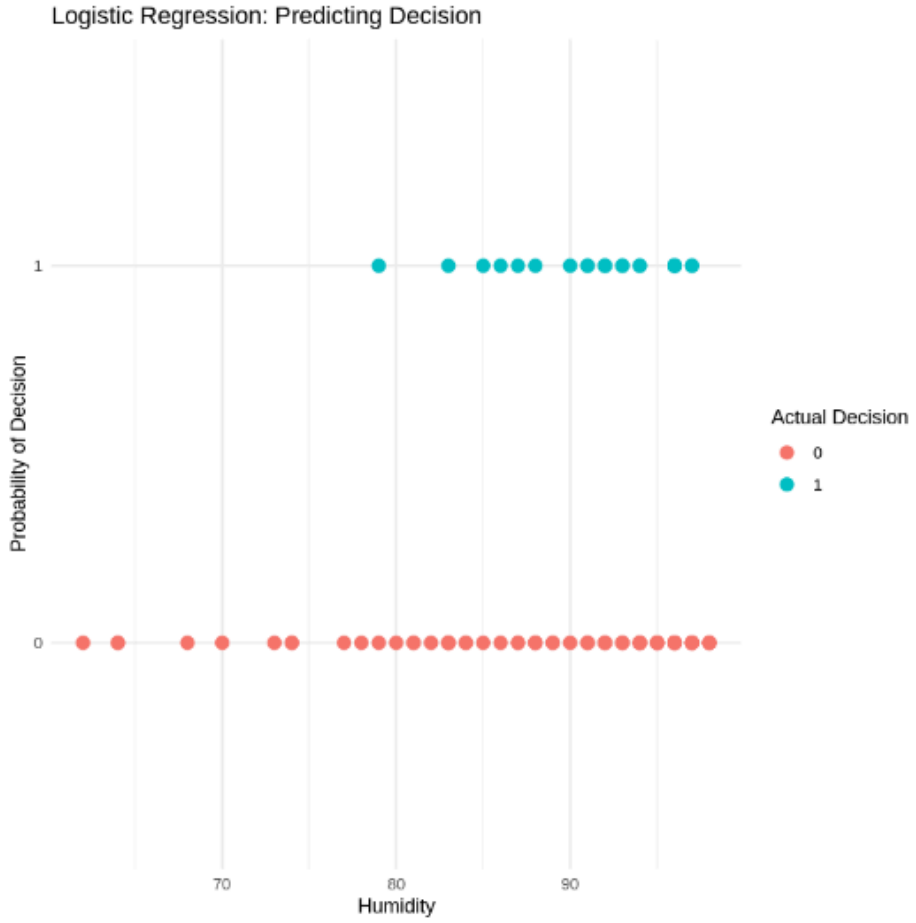
```
`geom_smooth()` using formula = 'y ~ x'
Warning message:
“Computation failed in `stat_smooth()`
Caused by error:
! y values must be 0 <= y <= 1”
```



Rains or not depending on temperature

```
[15] ggplot(data1, aes(x = humidity, y = Decision)) +
       geom_point(aes(color = factor(Decision)), size = 3) +
       geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
       labs(title = "Logistic Regression: Predicting Decision",
            x = "Humidity",
            y = "Probability of Decision",
            color = "Actual Decision") +
       theme_minimal()
```

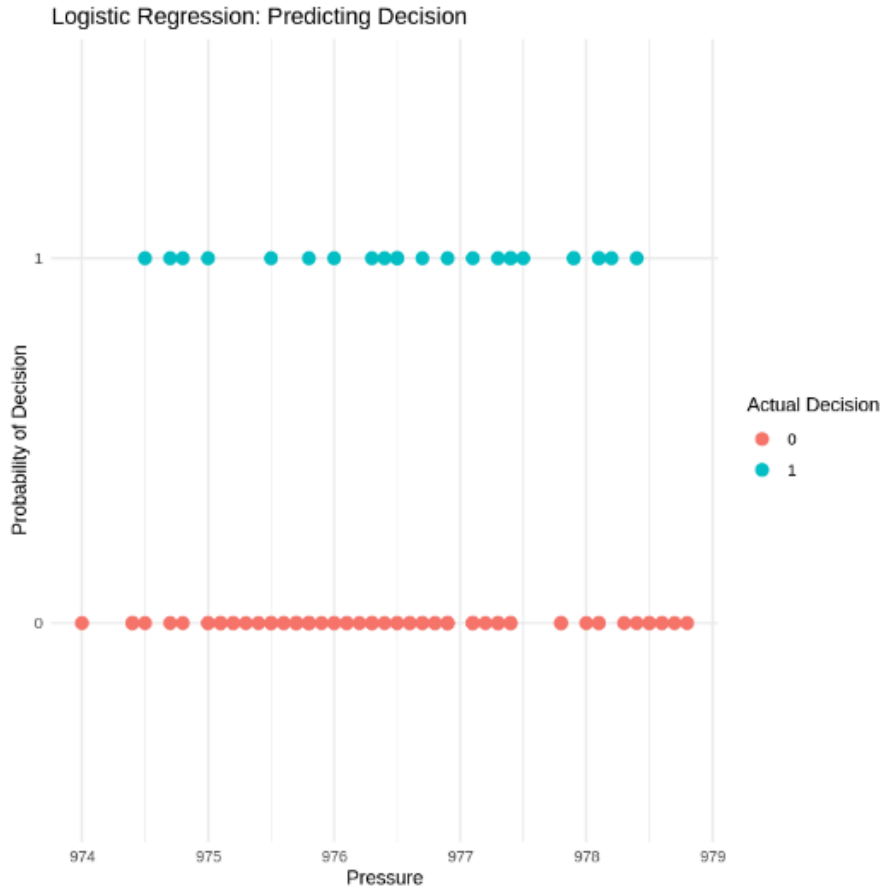
```
`geom_smooth()` using formula = 'y ~ x'
Warning message:
“Computation failed in `stat_smooth()`
Caused by error:
! y values must be 0 <= y <= 1”
```



Rains or not depending on humidity

```r
ggplot(data1, aes(x = Pressure, y = Decision)) +
  geom_point(aes(color = factor(Decision)), size = 3) +
  geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Logistic Regression: Predicting Decision",
       x = "Pressure",
       y = "Probability of Decision",
       color = "Actual Decision") +
  theme_minimal()
```

```
`geom_smooth()` using formula = 'y ~ x'
Warning message:
“Computation failed in `stat_smooth()`
Caused by error:
! y values must be 0 <= y <= 1”
```



Rains or not depending on pressure

D)Decision Tree with Feature Selection in R:-

Theory:
Decision Trees are useful supervised Machine learning algorithms that have the ability to perform both regression and classification tasks. It is characterized by nodes and branches, where the tests on each attribute are represented at the nodes, the outcome of this procedure is represented at the branches and the class labels are represented at the leaf nodes. Hence it uses a tree-like model based on various decisions that are used to compute their probable outcomes. These types of tree-based algorithms are one of the most widely used algorithms due to the fact that these algorithms are easy to interpret and use. Apart from this, the predictive models developed by this algorithm are found to have good stability and a descent accuracy due to which they are very popular.

Working of Decision Tree in R:

- Partitioning: It refers to the process of splitting the data set into subsets. The decision of making strategic splits greatly affects the accuracy of the tree. Many algorithms are used by the tree to split a node into sub-nodes which results in an overall increase in the clarity of the node with respect to the target variable. Various Algorithms like the chi-square and Gini index are used for this purpose and the algorithm with the best efficiency is chosen.
- Pruning: This refers to the process wherein the branch nodes are turned into leaf nodes which results in the shortening of the branches of the tree. The essence behind this idea is that overfitting is avoided by simpler trees as most complex classification trees may fit the training data well but do an underwhelming job in classifying new values.
- Selection of the tree: The main goal of this process is to select the smallest tree that fits the data due to the reasons discussed in the pruning section.

Feature Selection:

Feature selection in the context of decision trees involves choosing a subset of relevant features from the original set of variables to improve the model's performance, interpretability, and efficiency. Decision trees are sensitive to the choice of features, and including irrelevant or redundant variables can lead to overfitting, increased complexity, and decreased generalization.

Here's a brief overview of feature selection in decision trees in R:

- Importance of Feature Selection: Decision trees have a tendency to overfit the training data, capturing noise and details that may not generalize well to new data. Feature selection helps mitigate this issue by focusing on the most informative variables.

- Variable Importance Measures: Decision trees generate a variable importance score for each feature based on metrics such as Gini impurity or information gain. These scores reflect the contribution of each variable to the model's ability to make accurate predictions.

Implementing Decision Tree with Feature Selection in R:-

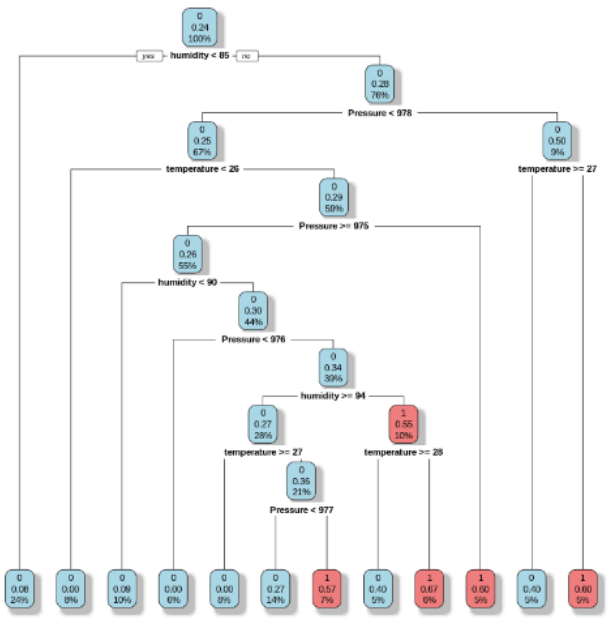
```
[ ]  install.packages("rpart")
     install.packages("rpart.plot")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)

Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
✓ 0s  [6]  data1<-read.csv("/content/rainfall_pred.csv", fileEncoding = "Latin1", check.names = F)
```

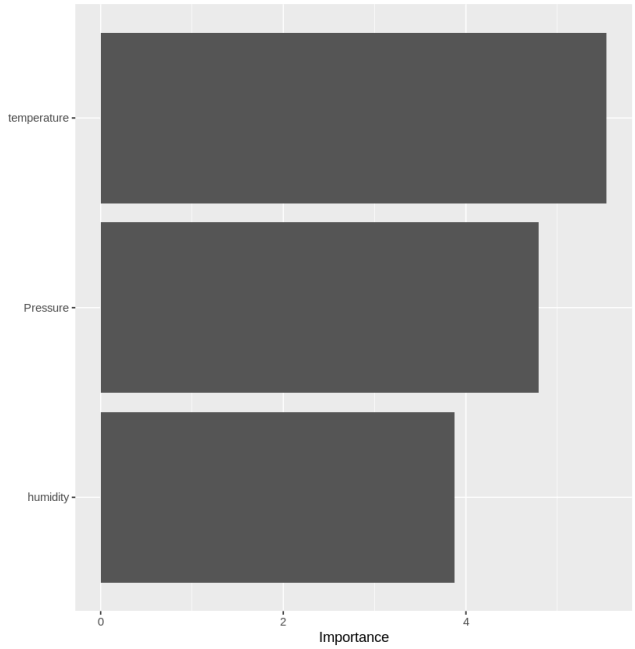
```
[ ]  tree_model <- rpart(Decision ~ temperature + humidity + Pressure, data = data1, method = "class",
                         minsplit = 10, minbucket = 5)

     # Print the complexity parameter table, which includes variable importance
     printcp(tree_model)

     # Plot the decision tree
     rpart.plot(tree_model, box.palette = c("lightblue", "lightcoral"), shadow.col = "gray")
```

```
Classification tree:
rpart(formula = Decision ~ temperature + humidity + Pressure,
    data = data1, method = "class", minsplit = 10, minbucket = 5)

Variables actually used in tree construction:
[1] humidity    Pressure    temperature

Root node error: 25/106 = 0.23585

n= 106

     CP nsplit rel error xerror    xstd
1 0.016      0       1.0   1.00 0.17483
2 0.010     11       0.8   1.56 0.19860
```



The importance of a variable can be quantified by the reduction in the impurity measure (e.g., Gini index or mean squared error) it brings when used for splitting.

```r
# Load the necessary library
library(vip)

# Create a variable importance plot
var_importance <- vip(tree_model, num_features = 10)
print(var_importance)
```



variable importance plot

Decision Tree with data processing and feature selection

Handling missing values

```r
# Load the necessary library
library(dplyr)

# Read the CSV file
data <- read.csv("/content/rainfall_pred.csv", fileEncoding = "Latin1", check.names = FALSE)

# Check the structure of the data
str(data)

# Check for missing values
sum(is.na(data))

# Summary statistics
summary(data)

# Handle missing values (if any)
data <- na.omit(data)  # remove rows with missing values
```

```
'data.frame':   106 obs. of  5 variables:
 $ temperature  : num  25.8 24.3 25.3 27.8 28.9 29.8 30.7 31.4 32 32.9 ...
 $ humidity     : int  91 96 93 83 80 77 73 70 68 64 ...
 $ precipitation: num  0.1 0.9 0.4 0.1 0 0.2 0.1 0.1 0.1 0.1 ...
 $ Pressure     : num  977 978 978 978 979 ...
 $ Decision     : int  0 1 0 0 0 0 0 0 0 0 ...
0
  temperature       humidity     precipitation        Pressure
 Min.   :24.30   Min.   :62.00   Min.   : 0.0000   Min.   :974.0
 1st Qu.:26.30   1st Qu.:85.25   1st Qu.: 0.0000   1st Qu.:975.7
 Median :27.20   Median :92.50   Median : 0.2000   Median :976.5
 Mean   :27.46   Mean   :89.65   Mean   : 0.8255   Mean   :976.5
 3rd Qu.:28.20   3rd Qu.:96.00   3rd Qu.: 0.7000   3rd Qu.:977.3
 Max.   :32.90   Max.   :98.00   Max.   :14.8000   Max.   :978.8
    Decision
 Min.   :0.0000
 1st Qu.:0.0000
 Median :0.0000
 Mean   :0.2358
 3rd Qu.:0.0000
 Max.   :1.0000
```

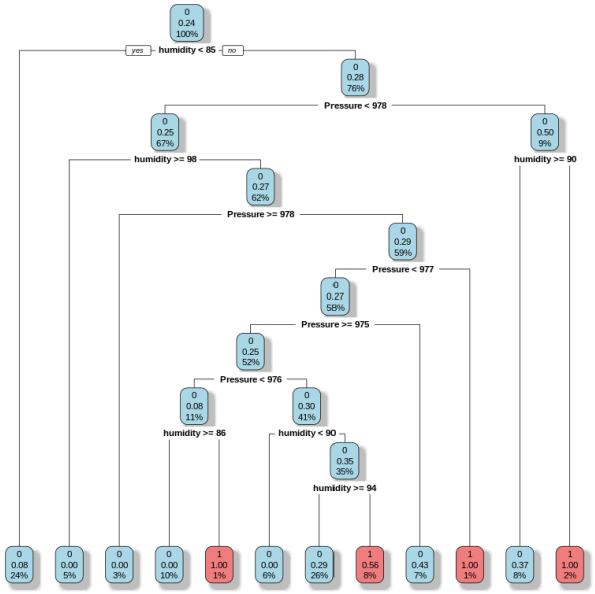
Feature selection

```r
#feature selection
library(caret)
column_to_remove <- "precipitation"

# Remove the specified column
data <- data[, !names(data) %in% column_to_remove]
# Calculate correlation matrix

cor_matrix <- cor(data)

# Find highly correlated features
highly_correlated <- findCorrelation(cor_matrix, cutoff = 0.8)

# Remove highly correlated features
data <- data[, -highly_correlated]
names(data)
```

'humidity' · 'precipitation' · 'Pressure' · 'Decision'

Fitting Decision tree

```r
library(rpart)
library(rpart.plot)
tree_model <- rpart(Decision ~ humidity + Pressure, data = data, method = "class",
                    minsplit = 10, minbucket = 0)

# Print the complexity parameter table, which includes variable importance
printcp(tree_model)

# Plot the decision tree
rpart.plot(tree_model, box.palette = c("lightblue", "lightcoral"), shadow.col = "gray")
```



Importance of Features

```
library(vip)

# Create a variable importance plot
var_importance <- vip(tree_model, num_features = 10)
print(var_importance)
```