NETAJI SUBHAS UNIVERSITY OF TECHNOLOGY

STATE UNIVERSITY UNDER DELHI ACT 06 OF 2018, GOVT. OF NCT OF DELHI
Azad Hind Fauji Marg, Sector-3, Dwarka, New Delhi-110078

Project on R

Computer Hardware and Software

(COCSC19)

SUBMITTED BY:
NAME: Ashmeet singh, Shubham Yadav, Himanshu Kumar
ROLL NO.: 2021UCS1639, 2021UCS1613,2021UCS4012
SEM/ BRANCH/ SEC: SEM 6 CSE 2

Q-1 : *Explain the basic data structure in R :*

In R, a programming language and environment for statistical computing and graphics, there are several basic data structures that are commonly uses. These data structures are fundamental for organizing and manipulating data efficiently. Here are some of the basic data structures in R.

1) Vectors:

A vector is a one-dimensional array that can hold elements of the same data type.

Elements in a vector can be numeric, character, logical, etc.

You can create a vector using the c() function.

```
# Creating a numeric vector
numeric_vector <- c(1, 2, 3, 4, 5)

# Creating a character vector
character_vector <- c("apple", "banana", "orange")
```

2) Matrices:

A matrix is a two-dimensional array with rows and columns.

All elements in a matrix must be of the same data type.

You can create a matrix using the matrix() function.

```
# Creating a matrix
mat <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3)
```

3) Lists:

A list is a collection of different data types and structures.

Elements in a list can be vectors, matrices, data frames, etc.

You can create a list using the list() function.

```r
# Creating a list
my_list <- list(
  numeric_vector = c(1, 2, 3),
  character_vector = c("a", "b", "c"),
  matrix = matrix(1:4, nrow = 2),
  data_frame = data.frame(Name = c("John", "Jane"), Age = c(28, 22)
)
```

4) Data Frames:

A data frame is a two-dimensional tabular data structure similar to a matrix, but with additional features.

Columns can have different data types, making it suitable for representing datasets.

You can create a data frame using the data.frame() function.

```r
# Creating a data frame
df <- data.frame(
  Name = c("Alice", "Bob", "Charlie"),
  Age = c(25, 30, 22),
  Score = c(85, 92, 78)
)
```

5) Arrays :

An Array is a multi dimensional extension of a matrix, It can have more than two dimensions and we can create array by using array() function.

```
# Creating a 3-dimensional array
arr <- array(c(1, 2, 3, 4, 5, 6, 7, 8, 9), dim = c(3, 3, 1))
```

.

These are some of the basic data structures in R. Understanding these structures is curcial for effective data manipulation and analysis in R. Additionally, R provides various functions and operations to perform operations on these data structures.

Q-2 Implement Linear Regression in R and Visualize the results.

Linear regression : It is a statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. The goal is to find the best-fitting line (or hyperplane in the case of multiple independent variables) that minimizes the sum of squared differences between the observed and predicted values.

DATASET SUMMARY
We have used car failure dataset which consists of 1624 entries and for each entry we have 7 variables (vehicle, fm, mileage, lh, lc, mc, state). Below is summary as summarized by R Studio.

Data	
car	51 obs. of 2 variables
data	1624 obs. of 7 variables
p1	List of 8
p2	List of 8
p3	List of 8
p4	List of 8
vehicle	1624 obs. of 7 variables
Values	
mycolors	chr [1:4] "blue" "#FFC125" "darkg…

DATA
A sample of 21 observations out of 1624 observation :

	Vehicle	fm	Mileage	lh	lc	mc	State
1	1	0	863	1.1	66.30	697.23	MS
2	2	10	4644	2.4	233.03	119.66	CA
3	3	15	16330	4.2	325.08	175.46	WI
4	4	0	13	1.0	66.64	0.00	OR
5	5	13	22537	4.5	328.66	175.46	AZ
6	6	21	40931	3.1	205.28	175.46	FL
7	7	11	34762	0.7	49.17	145.20	LA
8	8	5	11051	2.9	208.80	270.04	GA
9	9	8	7003	3.4	212.06	119.66	WA
10	10	1	11	0.7	44.43	0.00	PA
11	11	17	24879	3.5	260.29	119.66	TX
12	12	3	5339	3.2	236.93	440.13	LA
13	13	14	29782	10.0	695.10	228.12	FL
14	14	19	56111	2.0	116.00	183.31	OH
15	15	13	21946	3.8	312.36	175.46	MA
16	16	8	3101	3.1	220.61	119.66	VA
17	17	15	41965	0.9	66.25	119.66	OH
18	18	3	15365	2.0	158.94	175.46	CO
19	19	12	44865	4.9	319.51	119.66	FL
20	20	8	14025	1.4	87.42	1.85	NH
21	21	18	29987	2.6	182.17	128.21	IN

Code with outputs (had to direct copy code as my laptop does'nt support screenshot)

```
title: "Untitled"
author: "Supercow"
date: '2022-04-05'
output:
flexdashboard::flex_dashboard:
orientation: rows
vertical_layout: fill
---
```

```
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
```{r setup, include=FALSE}
library(flexdashboard)
library(knitr)
library(DT)
library(ggplot2)
library(plotly)
library(dplyr)
library(openintro)
library(highcharter)
library(ggvis)
```
```{r}
data <- read.csv("VehicleFailure.csv")
```
```{r}
mycolors <- c("blue", "#FFC125", "darkgreen", "darkorange"

```
Data Visualization
Row
Car Failure Analysis
Car Failures in US
```{r}
length(data$State)
```
[1] 1624
Labor Cost
```{r}
round(mean(data$lc),digits = 2)
```
[1] 242.92
Massachusetts
```{r}
sum(data$State == "MA")
```
[1] 21
California
```{r}
sum(data$State == "CA")

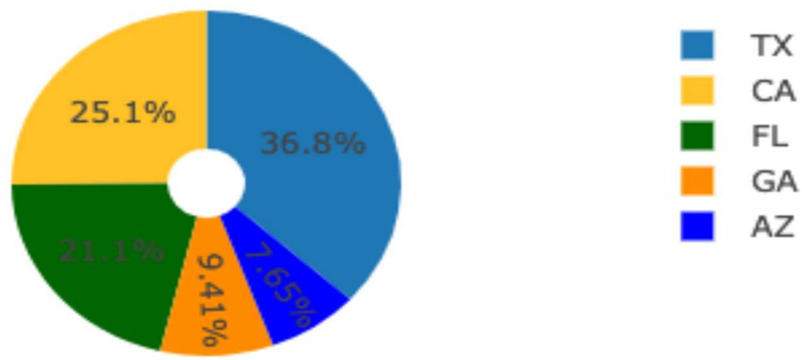
)

[1] 200
```
```

~~~
### Texas
```{r}
sum(data$State == "TX")
```
[1] 293
### Florida
```{r}
sum(data$State == "FL")
```
[1] 168
## Row
### Failures By State
```{r}
p1 <- data %>%
group_by(State) %>%
summarise(count = n()) %>%
plot_ly(x = ~State,
y = ~count,
color = "blue",
type = 'bar') %>%
layout(xaxis = list(title = "Failures By State"),
yaxis = list(title = 'Count'))
p1
~~~

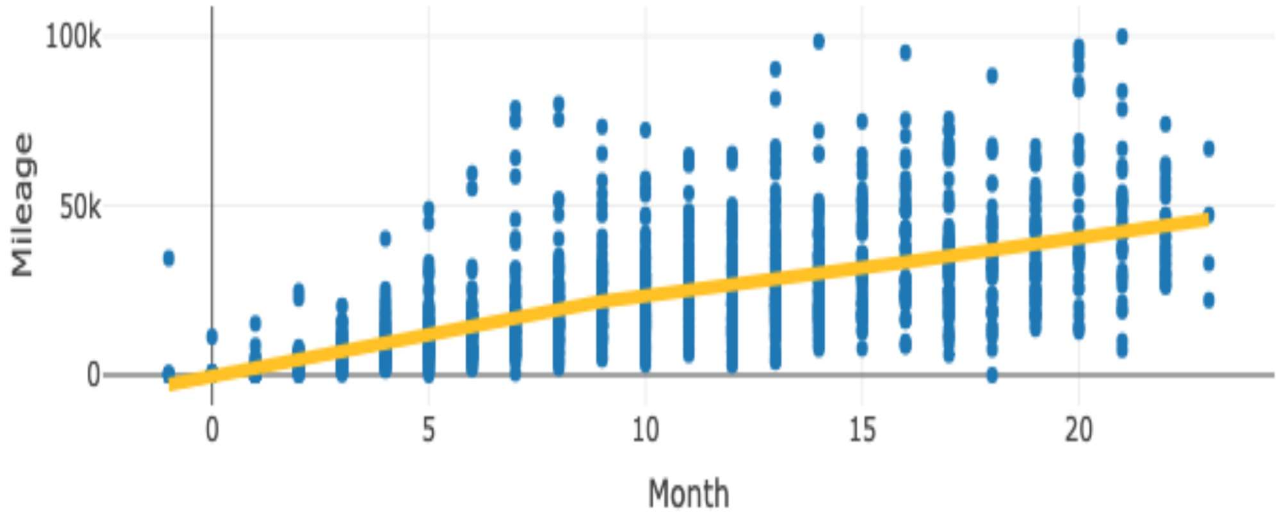

~~~
### Top States
```{r}
p2 <- data %>%
group_by(State) %>%
summarise(count = n()) %>%
~~~

```
filter(count>50) %>%
plot_ly(labels = ~State,
values = ~count,
marker = list(colors = mycolors)) %>%
add_pie(hole = 0.2) %>%
layout(xaxis = list(zeroline = F,
showline = F,
showticklabels = F,
showgrid = F),
yaxis = list(zeroline = F,
showline = F,
showticklabels=F,
showgrid=F))
p2
```



Scatter Plot of Month Vs Mileage (Linear regression plot)

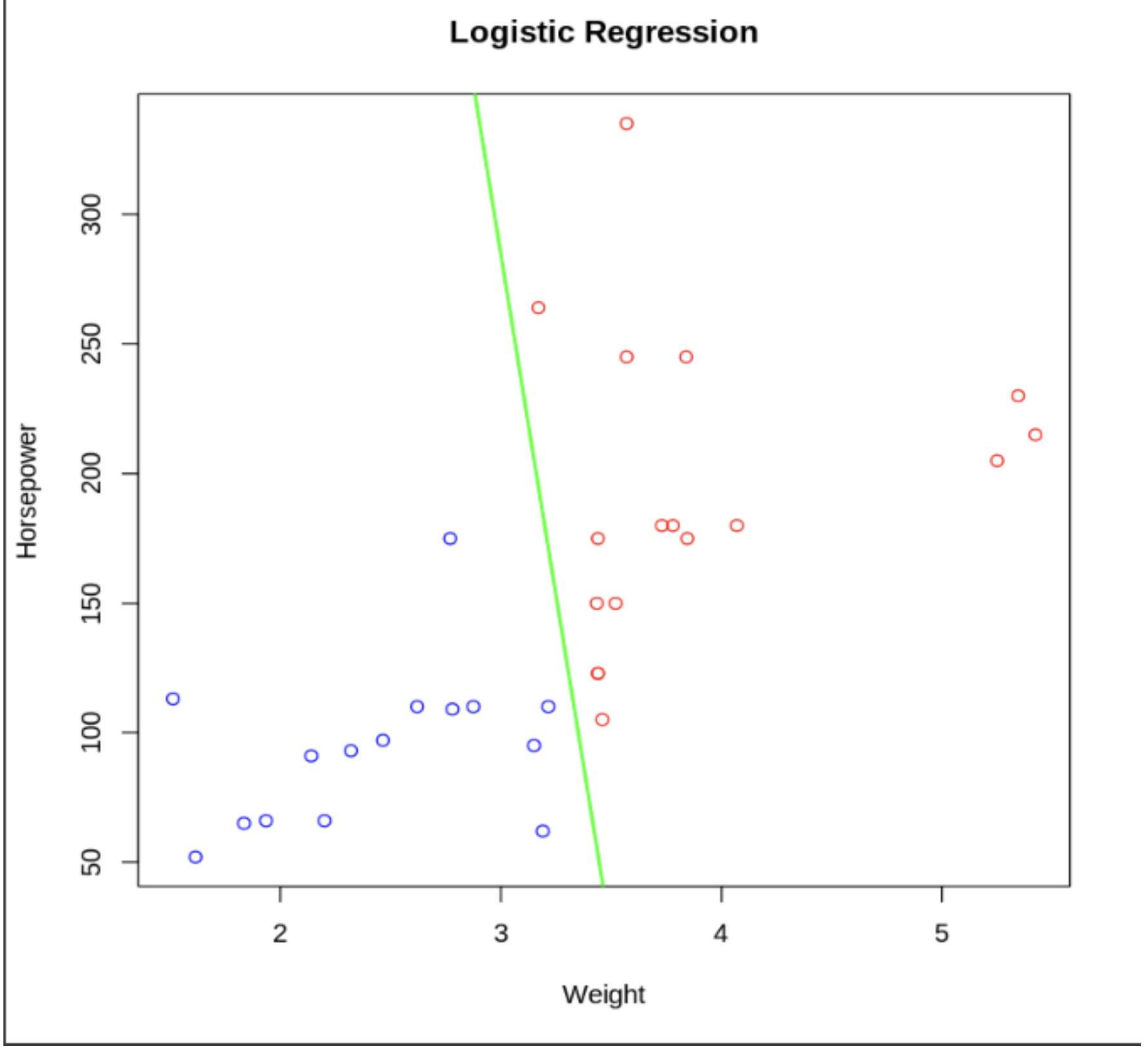
```{r}
p4 <- plot_ly(data, x=~fm) %>%
add_markers(y = ~Mileage,
text = ~paste("Mileage: ", Mileage),
showlegend = F) %>%
add_lines(y = ~fitted(loess(Mileage ~ fm)),
name = "Loess Smoother",
color = I("#FFC125"),
showlegend = T,
line = list(width=5)) %>%
layout(xaxis = list(title = "Month"),
yaxis = list(title = "Mileage"))
p4
```



3) Implement Logistic Regression in R and Visualize the results

Logistic regression : It is a statistical method used for modelling the probability of a binary outcome. It is particularly suitable for situations where the dependent variable is dichotomous (binary), meaning it has only two possible outcomes, typically coded as 0 and 1.

```r
# Loading the mtcars dataset
data(mtcars)

# Adding a binary response variable for binary classification
mtcars$high_mileage <- ifelse(mtcars$mpg > median(mtcars$mpg), 1, 0)

# Logistic regression model
logistic_model <- glm(high_mileage ~ wt + hp, data = mtcars, family = "binomial")

# Visualizing the results
plot(mtcars$wt, mtcars$hp, col = ifelse(mtcars$high_mileage == 1, "blue", "red"),
     main = "Logistic Regression", xlab = "Weight", ylab = "Horsepower")

# Adding decision boundary
boundary <- function(x) {
  return(-coef(logistic_model)[1] / coef(logistic_model)[3] - coef(logistic_model)[2] / coef(logistic_model)[3] * x)
}

x_values <- seq(min(mtcars$wt), max(mtcars$wt), length.out = 100)
lines(x_values, boundary(x_values), col = "green", lwd = 2)
```



Conclusion :

We have loaded the mtcars dataset and a binary response variable is created for binary classification. We are prediciting whether a car has high mileage or not. glm() function is used.

Visualizing the results: A scatter plot is created with blue points representing high mileage cars and red points representing low mileage cars. The logistic regression decision boundary is added to the plot in green

Q-4 : Implement any Machine learning Algorithm along with feature selection and data visualization on any dataset of your choice :

Following are the attributes and description :

Attribute	Description
Age	Age of the patient
Sex	Gender
thelach	Maximum heart rate
Rest_ecg	Electrocardiographic results Value 0: normal Value 1: having ST-T wave abnormality Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
fbs	Blood pressure (in mm Hg) Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
chol	Cholesterol in mg/dl fetched via BMI sensor
trtbps	Blood pressure (in mm Hg)
ca	Chest Pain type Value 1: typical angina Value 2: atypical angina

	Value 3: non-anginal pain Value 4: asymptomatic
exang	Exercise dis. [1 = yes; 0 = no]
target	0 = Less chance of heart attack 1 = More chance of heart attack

Loading the data and displaying its head :

heart_disease = read.csv("/Users/ashmeetsingh/Document/practical/CHS/archive/heart.csv")

head(heart_disease , 10)

Output:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
1	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
2	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
3	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
4	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
5	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
6	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
7	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
8	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
9	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
10	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1

>

Dividing dataset into subset with having ouput 1 and 0

target_1_data <- filter(heart_disease, output ==1) head(target_1_data)

target_0_data <- filter(heart_disease, output ==1)

head(target_0_data)

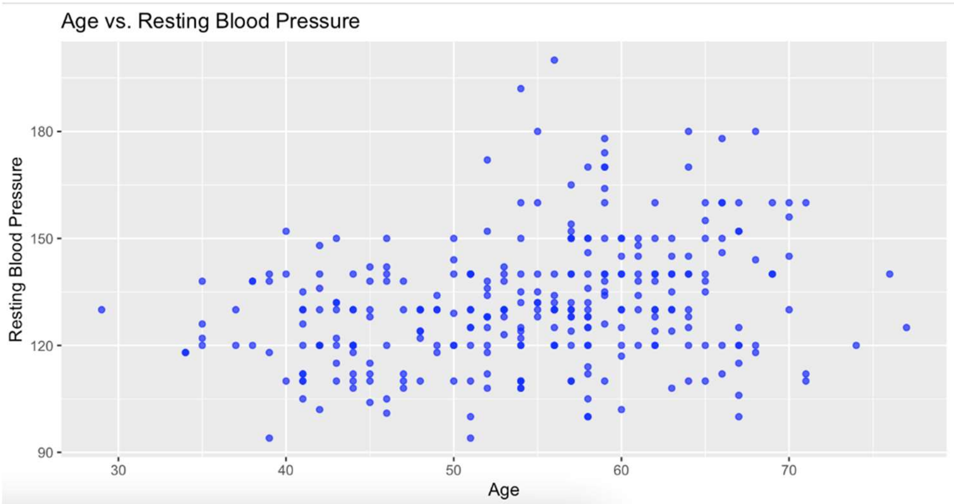
OUTPUT:

```
> head(target_1_data)
  age sex cp trtbps chol fbs restecg thalachh exng oldpeak slp caa thall output
1  63   1  3    145  233   1       0      150    0     2.3   0   0     1      1
2  37   1  2    130  250   0       1      187    0     3.5   0   0     2      1
3  41   0  1    130  204   0       0      172    0     1.4   2   0     2      1
4  56   1  1    120  236   0       1      178    0     0.8   2   0     2      1
5  57   0  0    120  354   0       1      163    1     0.6   2   0     2      1
6  57   1  0    140  192   0       1      148    0     0.4   1   0     1      1
> target_0_data <- filter(heart_disease, output == 0)
> head(target_0_data)
  age sex cp trtbps chol fbs restecg thalachh exng oldpeak slp caa thall output
1  67   1  0    160  286   0       0      108    1     1.5   1   3     2      0
2  67   1  0    120  229   0       0      129    1     2.6   1   2     3      0
3  62   0  0    140  268   0       0      160    0     3.6   0   2     2      0
4  63   1  0    130  254   0       0      147    0     1.4   1   1     3      0
5  53   1  0    140  203   1       0      155    1     3.1   0   0     3      0
6  56   1  2    130  256   1       0      142    1     0.6   1   1     1      0
>
```

Age vs Resting BP plot graph:

```
#Age vs Resting BP plot graph
ggplot(heart_disease, aes(x = age, y = trtbps)) +
  geom_point(color = "blue", alpha = 0.7) +
  labs(title = "Age vs. Resting Blood Pressure",
       x = "Age",
       y = "Resting Blood Pressure")
```

OUTPUT:



Density graph of age and heart rate :

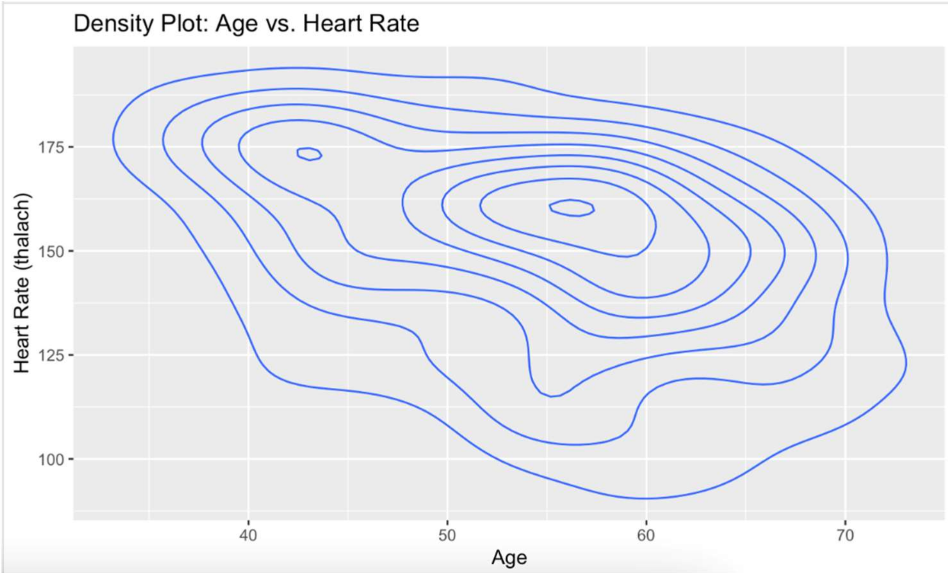
ggplot(heart_disease, aes(x= age, y = thalachh)) + geom_density_2d() +

labs(title = “Density plot:Age vs Heart rate”,

x= “Age”

y= “Heart Rate(thalach)”)

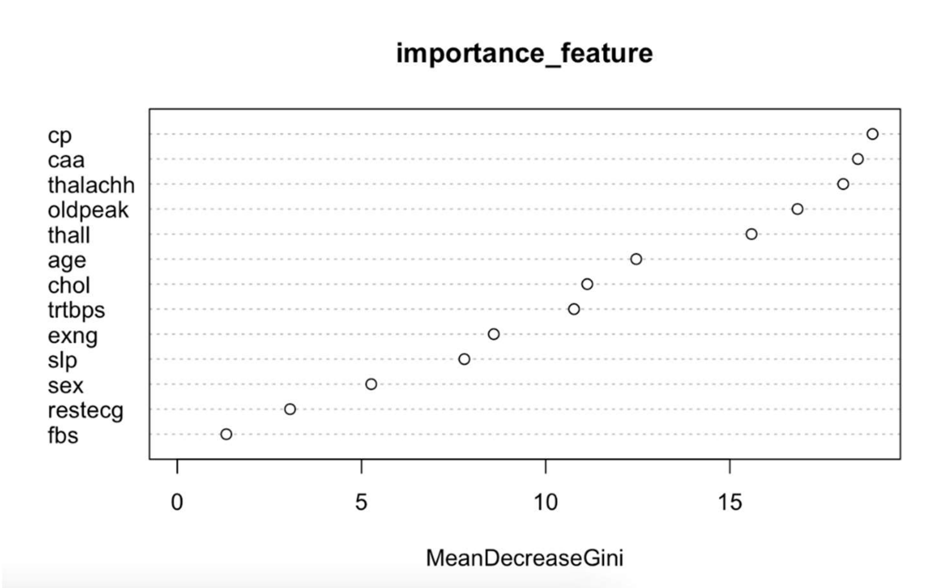
OUTPUT:



Random Forest

```
importance_feature <- randomForest(as.factor(output)~ age+ sex+ cp+ trtbps+ chol+ fbs+ restecg+ thalachh+ exng +oldpeak+ slp+ caa+ thall,data = heart_disease ,ntree=500 )
importance(importance_feature)
varImpPlot(importance_feature)
```

OUTPUT:



Histogram for heart attack v/s age :