Ex.No : 7

Implement Linear and Logistic Regression

AIM:

To implement linear and logistic regression techniques in machine learning.

PROCEDURES:

Linear Regression

1. Define vectors for heights and weights.
2. Combine the heights and weights into a data frame.
3. Fit a linear regression model using height to predict weight.
4. Print the summary of the linear regression model to view model statistics.
5. Open a new graphical device for plotting.
6. Create a scatter plot of height vs. weight data points.
7. Label the plot with a title, x-axis label (Height), and y-axis label (Weight).
8. Set plot points with specific color (blue) and style (solid circle).
9. Add the fitted linear regression line to the plot.
10. Customize the regression line with red color and a thicker width.

Logistic Regression

1. Load the `mtcars` dataset.
2. Convert the `am` column from numeric to a factor with labels "Automatic" and "Manual."
3. Fit a logistic regression model to predict `am` (transmission) based on `mpg` (miles per gallon).
4. Print the summary of the logistic regression model.
5. Predict the probabilities of manual transmission using the logistic model.
6. Print the predicted probabilities for manual transmission.
7. Create a scatter plot of `mpg` vs. transmission type (manual/automatic).
8. Label the plot with a title, x-axis label (MPG), and y-axis label (Probability of Manual Transmission).

9. Set plot points with blue color and solid circles.
10. Add the logistic regression curve to the plot, colored red with a thicker line.

CODE:

LinearRegression.py

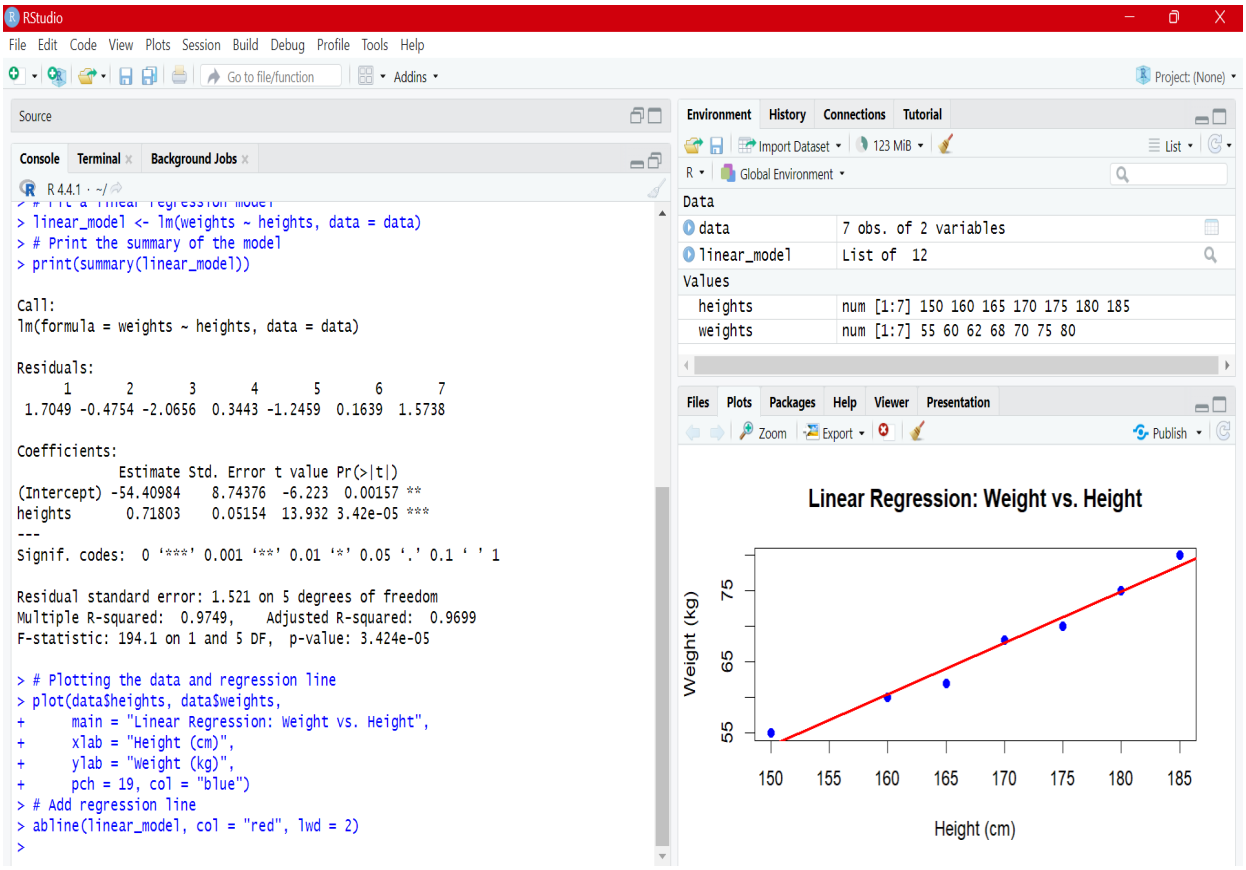
```
# Sample data
heights <- c(150, 160, 165, 170, 175, 180, 185)
weights <- c(55, 60, 62, 68, 70, 75, 80)

# Create a data frame
data <- data.frame(heights, weights)

# Fit a linear regression model
linear_model <- lm(weights ~ heights, data = data)

# Print the summary of the model
print(summary(linear_model))

# Plotting the data and regression line
dev.new()
plot(data$heights, data$weights,
    main = "Linear Regression: Weight vs. Height",
    xlab = "Height (cm)",
    ylab = "Weight (kg)",
    pch = 19, col = "blue")

# Add regression line
abline(linear_model, col = "red", lwd = 2)
```

LogisticRegression.py

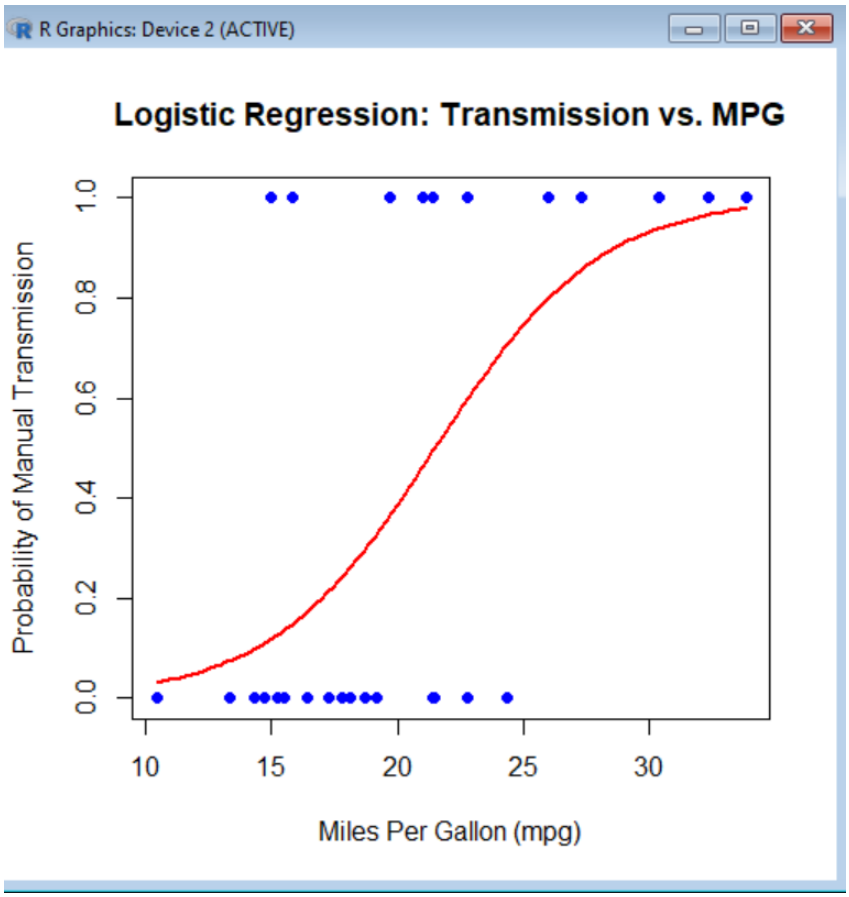
```
# Load the dataset
```

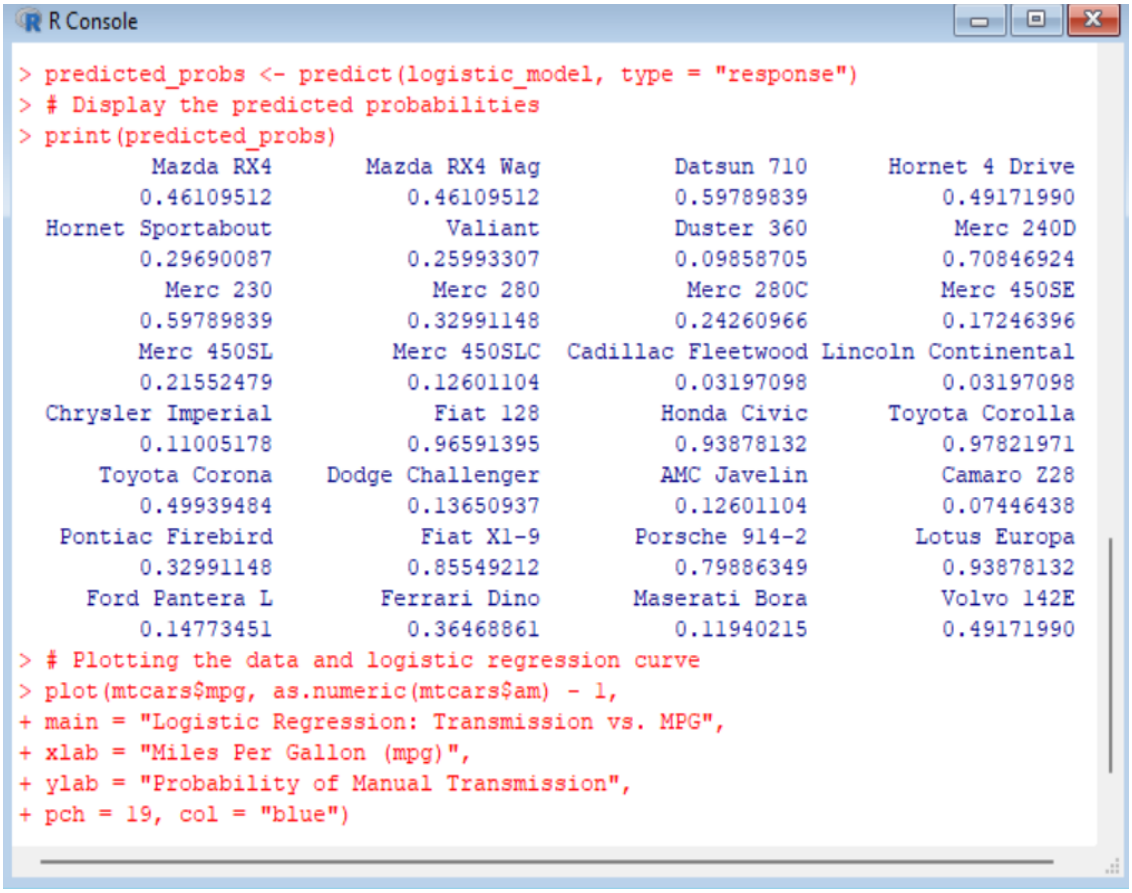
```
data(mtcars)
# Convert 'am' to a factor (categorical variable)
mtcars$am <- factor(mtcars$am, levels = c(0, 1),
                    labels = c("Automatic", "Manual"))
# Fit a logistic regression model
logistic_model <- glm(am ~ mpg, data = mtcars, family = binomial)
# Print the summary of the model
print(summary(logistic_model))
# Predict probabilities for the logistic model
predicted_probs <- predict(logistic_model, type = "response")
# Display the predicted probabilities
print(predicted_probs)
# Plotting the data and logistic regression curve
plot(mtcars$mpg, as.numeric(mtcars$am) - 1,
     main = "Logistic Regression: Transmission vs. MPG",
     xlab = "Miles Per Gallon (mpg)",
     ylab = "Probability of Manual Transmission",
     pch = 19, col = "blue")
# Add the logistic regression curve
curve(predict(logistic_model, data.frame(mpg = x), type = "response"),
      add = TRUE, col = "red", lwd = 2)
```

OUTPUT


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Source
Console Terminal Background Jobs
R 4.4.1 · ~/
> linear_model <- lm(weights ~ heights, data = data)
> # Print the summary of the model
> print(summary(linear_model))
Call:
lm(formula = weights ~ heights, data = data)
Residuals:
1 2 3 4 5 6 7
1.7049 -0.4754 -2.0656 0.3443 -1.2459 0.1639 1.5738
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) -54.40984 8.74376 -6.223 0.00157 **
heights 0.71803 0.05154 13.932 3.42e-05 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.521 on 5 degrees of freedom
Multiple R-squared: 0.9749, Adjusted R-squared: 0.9699
F-statistic: 194.1 on 1 and 5 DF, p-value: 3.424e-05
> # Plotting the data and regression line
> plot(data$heights, data$weights,
+ main = "Linear Regression: Weight vs. Height",
+ xlab = "Height (cm)",
+ ylab = "Weight (kg)",
+ pch = 19, col = "blue")
> # Add regression line
> abline(linear_model, col = "red", lwd = 2)
Environment History Connections Tutorial
Import Dataset 123 MiB
Global Environment
Data
data 7 obs. of 2 variables
linear_model List of 12
Values
heights num [1:7] 150 160 165 170 175 180 185
weights num [1:7] 55 60 62 68 70 75 80
Files Plots Packages Help Viewer Presentation
Zoom Export Publish
Linear Regression: Weight vs. Height
Weight (kg)
55
65
75
150
155
160
165
170
175
180
185
Height (cm)


R Graphics: Device 2 (ACTIVE)
Logistic Regression: Transmission vs. MPG
Probability of Manual Transmission
0.0
0.2
0.4
0.6
0.8
1.0
10
15
20
25
30
Miles Per Gallon (mpg)



RESULT:

Thus, to implement linear and logistic regression using machine learning is completed successfully.