Ex.No – 7 Roll No – 210701190

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 #

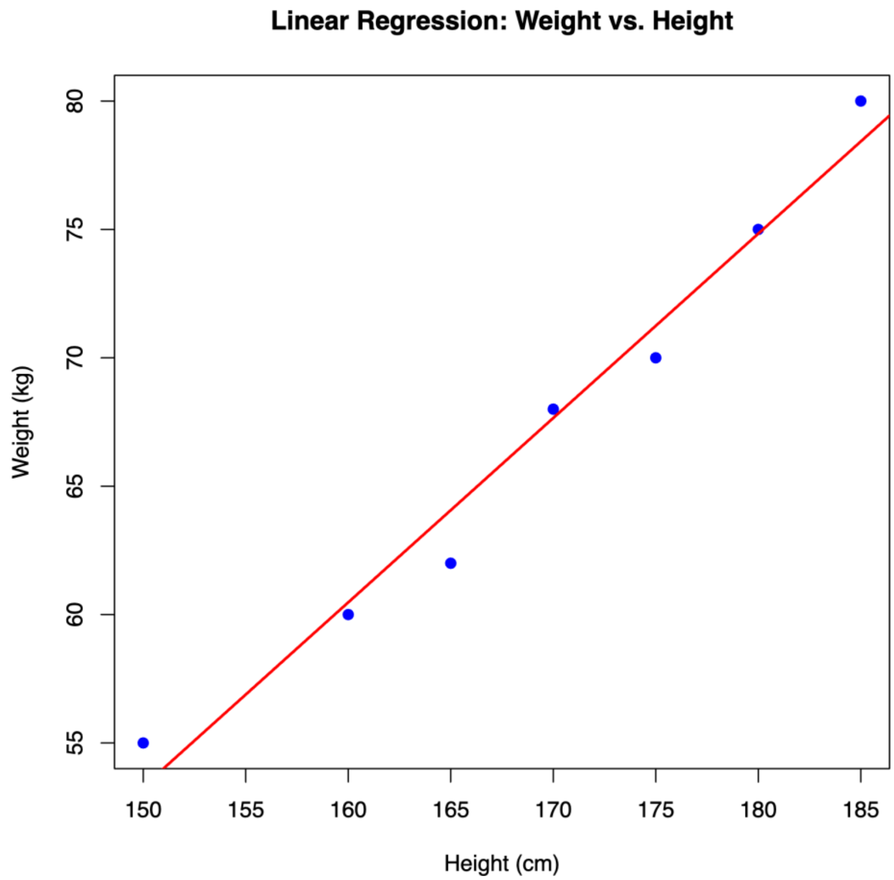
Sample data

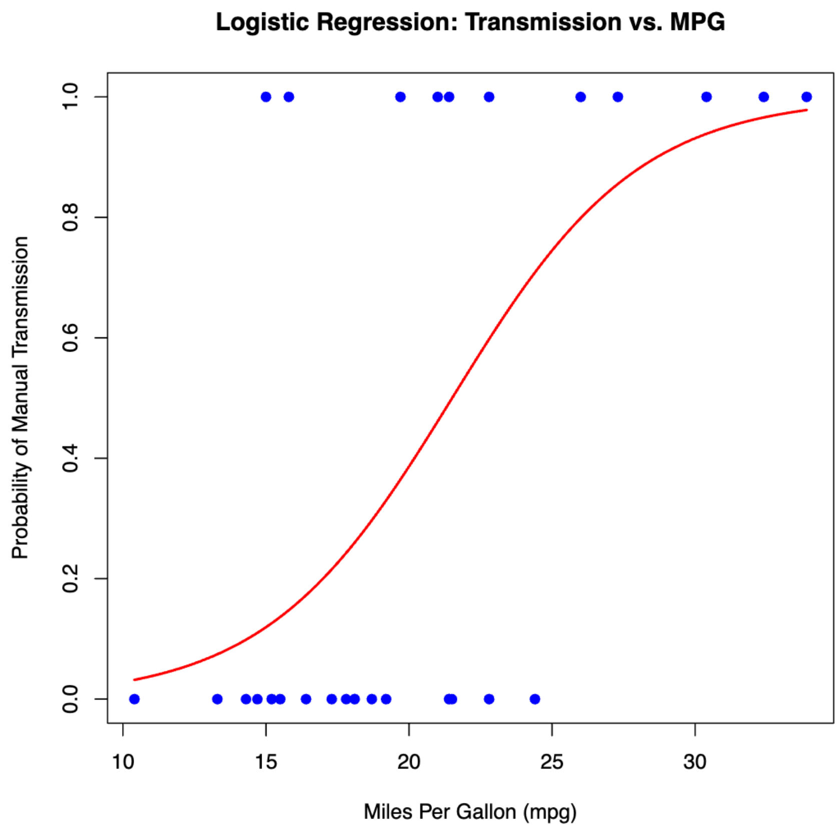
```
heights <- c(150, 160, 165, 170, 175, 180, 185)
weights <- c(55, 60, 62, 68, 70, 75, 80)

# 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:


Linear Regression: Weight vs. Height
Weight (kg)
80
75
70
65
60
55
150
155
160
165
170
175
180
185
Height (cm)



RESULT:

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