Ex No 7

Implement Linear and Logistic Regression in R

AIM:

To Implement Linear and Logistic Regression using R

PROCEDURE:

- Collect and load the dataset from sources like CSV files or databases.
- Clean and preprocess the data, including handling missing values and encoding categorical variables.
- Split the dataset into training and testing sets to evaluate model performance.
- Normalize or standardize the features to ensure consistent scaling. 5.Choose the appropriate model: Linear Regression for continuous outcomes.
- Train the model on the training data using the `fit` method.
- Make predictions on the testing data using the `predict` method.
- Evaluate the model using metrics like Mean Squared Error (MSE) for Linear Regression or accuracy and confusion matrix for Logistic Regression.
- Visualize the results with plots, such as scatter plots for Linear Regression or decision boundaries for Logistic Regression.
- Fine-tune the model by adjusting hyperparameters or applying regularization Techniques.

CODE:

LinearRegression.R:

```
# 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
plot(data$heights, data$weights,
    main = "Linear Regression: Weight vs. Height",
    xlab = "Height (cm)", ylab = "Weight (kg)", pch =
    19, col = "blue")
```

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

LogisticRegression.R:

```r
# 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)
```
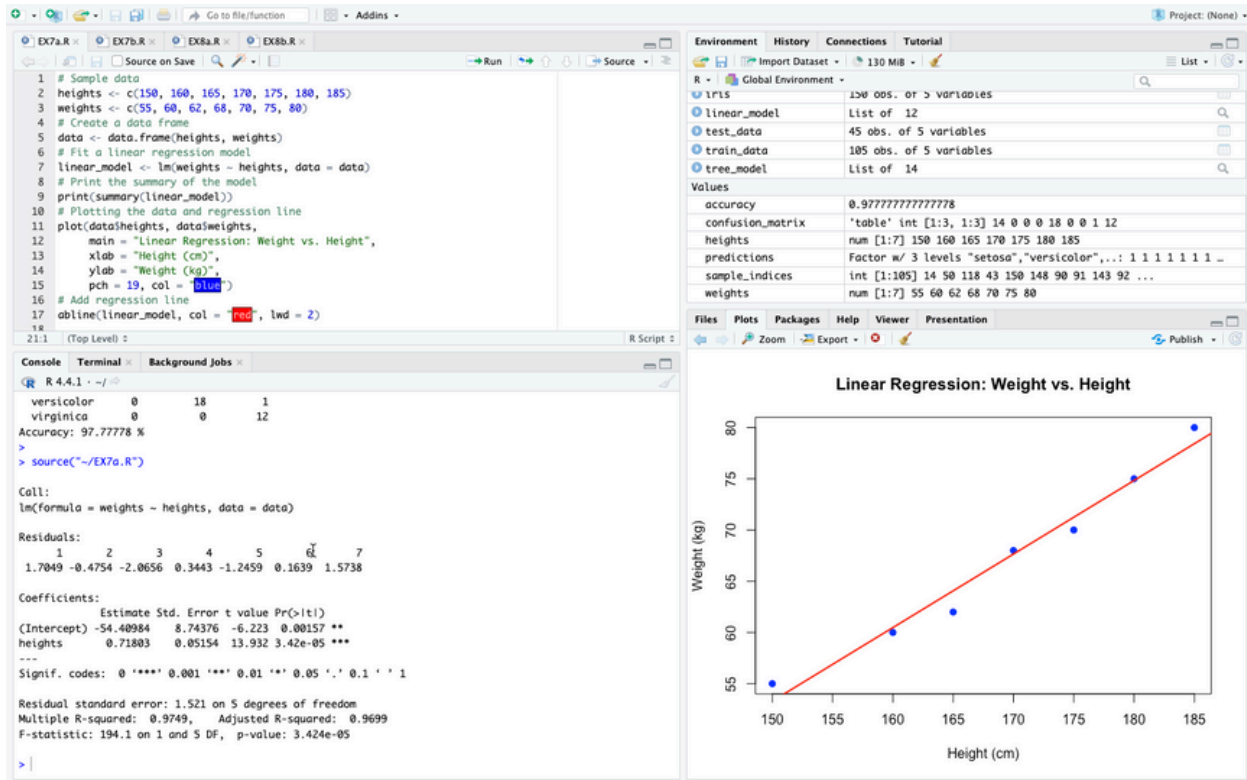
OUTPUT:

Linear Regression:


```r
1  # Sample data
2  heights <- c(150, 160, 165, 170, 175, 180, 185)
3  weights <- c(55, 60, 62, 68, 70, 75, 80)
4  # Create a data frame
5  data <- data.frame(heights, weights)
6  # Fit a linear regression model
7  linear_model <- lm(weights ~ heights, data = data)
8  # Print the summary of the model
9  print(summary(linear_model))
10 # Plotting the data and regression line
11 plot(data$heights, data$weights,
12     main = "Linear Regression: Weight vs. Height",
13     xlab = "Height (cm)",
14     ylab = "Weight (kg)",
15     pch = 19, col = "blue")
16 # Add regression line
17 abline(linear_model, col = "red", lwd = 2)
18
```
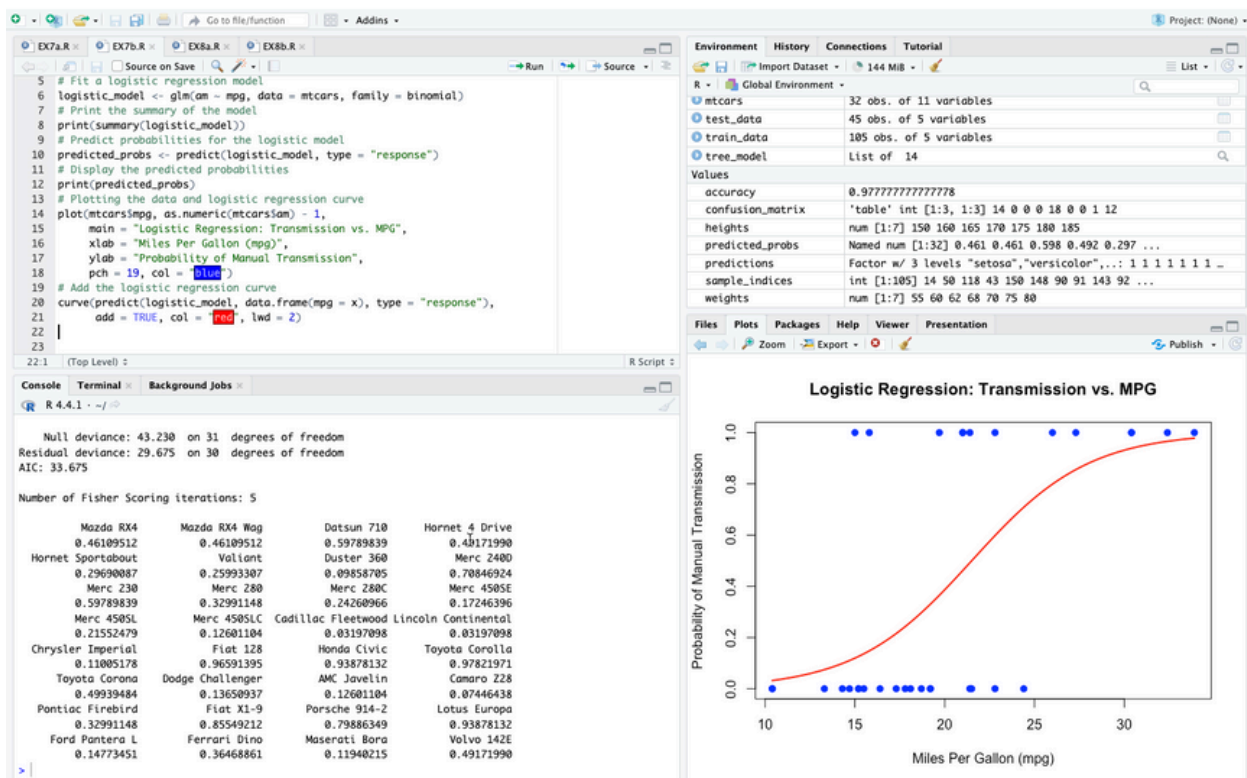
Console output:

```
versicolor      0       18       1
virginica       0        0      12
Accuracy: 97.77778 %
>
> source("~/EX7a.R")

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

>
```

Logistic Regression:


```r
5  # Fit a logistic regression model
6  logistic_model <- glm(am ~ mpg, data = mtcars, family = binomial)
7  # Print the summary of the model
8  print(summary(logistic_model))
9  # Predict probabilities for the logistic model
10 predicted_probs <- predict(logistic_model, type = "response")
11 # Display the predicted probabilities
12 print(predicted_probs)
13 # Plotting the data and logistic regression curve
14 plot(mtcars$mpg, as.numeric(mtcars$am) - 1,
15     main = "Logistic Regression: Transmission vs. MPG",
16     xlab = "Miles Per Gallon (mpg)",
17     ylab = "Probability of Manual Transmission",
18     pch = 19, col = "blue")
19 # Add the logistic regression curve
20 curve(predict(logistic_model, data.frame(mpg = x), type = "response"),
21     add = TRUE, col = "red", lwd = 2)
22
23
```

Console output:

```
    Null deviance: 43.230  on 31  degrees of freedom
Residual deviance: 29.675  on 30  degrees of freedom
AIC: 33.675

Number of Fisher Scoring iterations: 5

          Mazda RX4      Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
         0.46109512         0.46109512          0.59789839          0.40171990
   Hornet Sportabout            Valiant          Duster 360           Merc 240D
         0.29690087         0.25993307          0.09858705          0.70846924
           Merc 230           Merc 280           Merc 280C           Merc 450SE
         0.59789839         0.32991148          0.24260966          0.17246396
          Merc 450SL        Merc 450SLC  Cadillac Fleetwood Lincoln Continental
         0.21552479         0.12601104          0.03197098          0.03197098
   Chrysler Imperial           Fiat 128          Honda Civic       Toyota Corolla
         0.11005178         0.96591395          0.93878132          0.97821971
       Toyota Corona    Dodge Challenger          AMC Javelin          Camaro Z28
         0.49939484         0.13650937          0.12601104          0.07446438
     Pontiac Firebird          Fiat X1-9       Porsche 914-2        Lotus Europa
         0.32991148         0.85549212          0.79886349          0.93878132
      Ford Pantera L        Ferrari Dino        Maserati Bora           Volvo 142E
         0.14773451         0.36468861          0.11940215          0.49171990
>
```

RESULT:

Thus to Implement Linear and Logistic Regression using R has been successfully executed.