A PYTHON PROGRAM TO IMPLEMENT UNIVARIATE, BIVARIATE AND MULTIVARIATE REGRESSION

AIM:

To implement a python program using univariate, bivariate and multivariate regression features for a given iris dataset.

ALGORITHM:

Step 1: Import necessary libraries:

● Pandas for data manipulation, numpy for numerical operations,and matplotlib.pyplot for plotting.

Step 2: Read the dataset:

● Use the pandas `read_csv` function to read the dataset.

● Store the dataset in a variable (e.g., `data`).

Step 3: Prepare the data:

● Extract the independent variable(s) (X) and dependent variable (y) from the dataset.

● Reshape X and y to be 2D arrays if needed.

Step 4:Univariate Regression:

● For univariate regression, use only one independent variable.

● Fit a linear regression model to the data using numpy's polyfit function or sklearn's LinearRegression class.

● Make predictions using the model.

● Calculate the R-squared value to evaluate the model's performance.

Step 5: Bivariate Regression:

● For bivariate regression, use two independent variables.

● Fit a linear regression model to the data using numpy's `polyfit` function or sklearn's `LinearRegression` class.

● Make predictions using the model.

● Calculate the R-squared value to evaluate the model's performance.

Step 6: Multivariate Regression:

● For multivariate regression, use more than two independent variables.

● Fit a linear regression model to the data using sklearn's `LinearRegression` class.

● Make predictions using the model.

● Calculate the R-squared value to evaluate the model's performance.

Step 7: Plot the results:

● For univariate regression, plot the original data points (X, y) as a scatter plot and the regression line as a line plot.

● For bivariate regression, plot the original data points (X1, X2, y) as a 3D scatter plot and the regression plane.

● For multivariate regression, plot the predicted values against the actual values.

Step 8: Display the results:

● Print the coefficients (slope) and intercept for each regression model.

● Print the R-squared value for each regression model.

Step 9: Complete the program:

● Combine all the steps into a Python program.

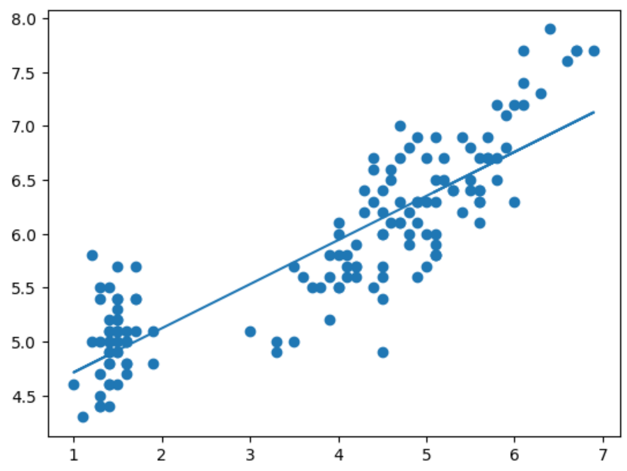
● Run the program to perform univariate, bivariate, and multivariate regression on the dataset.

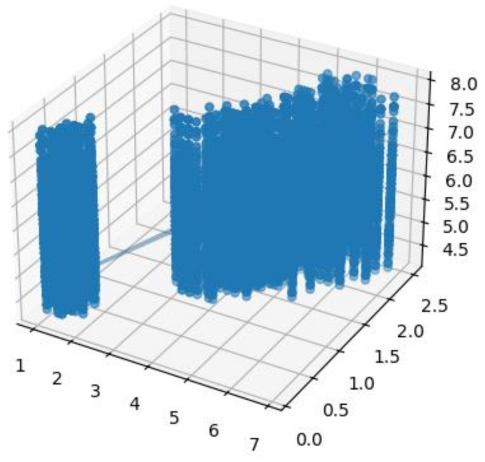
PROGRAM:

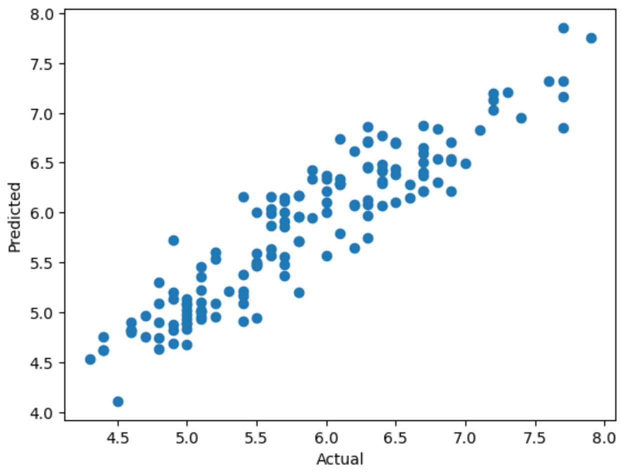
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression,LogisticRegression
from sklearn.metrics import r2_score,accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris
from mpl_toolkits.mplot3d import Axes3D

d1 = pd.read_csv("iris.csv")

y = d1["sepal.length"].values.reshape(-1,1)

x1 = d1["petal.length"].values.reshape(-1,1)
m1 = LinearRegression()
m1.fit(x1,y)
p1 = m1.predict(x1)
r1 = r2_score(y,p1)

plt.scatter(x1,y)
plt.plot(x1,p1)
plt.show()

x2 = d1[["petal.length","petal.width"]].values
m2 = LinearRegression()
m2.fit(x2,y)
p2 = m2.predict(x2)
r2 = r2_score(y,p2)

f = plt.figure()
a = f.add_subplot(111,projection="3d")
a.scatter(x2[:,0],x2[:,1],y)
```

```python
a.plot_trisurf(x2[:,0],x2[:,1],p2.flatten(),alpha=0.5)
plt.show()

x3 = d1.drop(["sepal.length", "variety"],axis=1).values
m3 = LinearRegression()
m3.fit(x3,y)
p3 = m3.predict(x3)
r3 = r2_score(y,p3)

plt.scatter(y,p3)
plt.xlabel("Actual")
plt.ylabel("Predicted")
plt.show()

print(m1.coef_,m1.intercept_,r1)
print(m2.coef_,m2.intercept_,r2)
print(m3.coef_,m3.intercept_,r3)

d2 = load_iris()
x = d2.data
y = d2.target

xtr,xts,ytr,yts = train_test_split(x,y,test_size=0.2,random_state=42)

m = LogisticRegression(max_iter=200)
m.fit(xtr,ytr)

p = m.predict(xts)
a = accuracy_score(yts,p)
print(a*100)

print("Enter flower measurements:")
s1 = float(input("Sepal Length: "))
s2 = float(input("Sepal Width: "))
p1 = float(input("Petal Length: "))
p2 = float(input("Petal Width: "))

u = [[s1,s2,p1,p2]]
r = m.predict(u)

print("Predicted Iris Flower:",d2.target_names[r[0]])
```

OUTPUT:







```
[[0.40892228]] [4.30660342] 0.759954645772515
[[ 0.54177715 -0.31955056]] [4.19058243] 0.7662612975425306
[[ 0.65083716  0.70913196 -0.55648266]] [1.85599749] 0.8586117200663177
100.0
Enter flower measurements:
Sepal Length: 10
Sepal Width: 20
Petal Length: 15
Petal Width: 25
Predicted Iris Flower: virginica
```

RESULT:

Thus, the merged Python program successfully performs univariate, bivariate, and multivariate regression analysis and accurately predicts the Iris flower species using Logistic Regression based on user input.