EXPERIMENT 1:

A PYTHON PROGRAM USING 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.

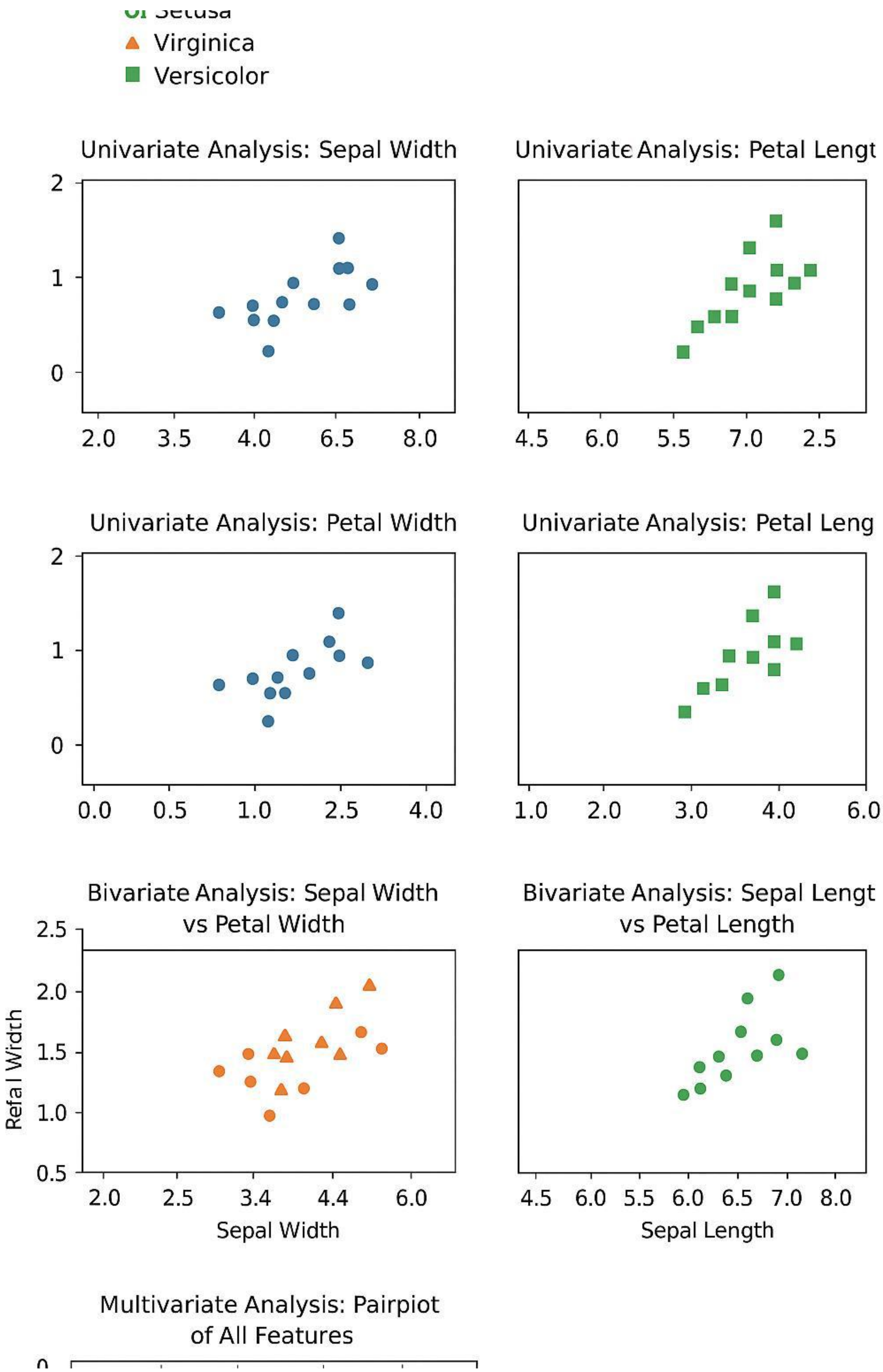
Step 8: Display the results:
- Print the coefficients (slope) and intercept for each regression model.
- Print the R-squared value for each regression model.

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.

CODE:-

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
df = pd.read_csv('../input/iris-dataset/iris.csv') df.head(150)
df.shape (150,5)

#univariate for sepal width
df.loc[df['variety']=='Setosa'] df_Setosa=df.loc[df['variety']=='Setosa']
df_Virginica=df.loc[df['variety']=='Virginica'] df_Versicolor=df.loc[df['variety']=='Versicolor']
plt.scatter(df_Setosa['sepal.width'],np.zeros_like(df_Setosa['sepal.width']))
plt.scatter(df_Virginica['sepal.width'],np.zeros_like(df_Virginica['sepal.width']))
plt.scatter(df_Versicolor['sepal.width'],np.zeros_like(df_Versicolor['sepal.width']))
plt.xlabel('sepal.width')
plt.show()

#univariate for sepal length
df.loc[df['variety']=='Setosa']
```

```
df_Setosa=df.loc[df['variety']=='Setosa']
df_Virginica=df.loc[df['variety']=='Virginica']
df_Versicolor=df.loc[df['variety']=='Versicolor']
plt.scatter(df_Setosa['sepal.length'],np.zeros_like(df_Setosa['sepal.length']))
plt.scatter(df_Virginica['sepal.length'],np.zeros_like(df_Virginica['sepal.length']))
plt.scatter(df_Versicolor['sepal.length'],np.zeros_like(df_Versicolor['sepal.length'])) plt.xlabel('sepal.length')
plt.show()

#univariate for petal width
df.loc[df['variety']=='Setosa']
df_Setosa=df.loc[df['variety']=='Setosa']
df_Virginica=df.loc[df['variety']=='Virginica']
df_Versicolor=df.loc[df['variety']=='Versicolor']
plt.scatter(df_Setosa['petal.width'],np.zeros_like(df_Setosa['petal.width']))
plt.scatter(df_Virginica['petal.width'],np.zeros_like(df_Virginica['petal.width']))
plt.scatter(df_Versicolor['petal.width'],np.zeros_like(df_Versicolor['petal.width']))
plt.xlabel('petal.width')
plt.show()


#univariate for petal length
df.loc[df['variety']=='Setosa'] df_Setosa=df.loc[df['variety']=='Setosa']
df_Virginica=df.loc[df['variety']=='Virginica'] df_Versicolor=df.loc[df['variety']=='Versicolor']
plt.scatter(df_Setosa['petal.length'],np.zeros_like(df_Setosa['petal.length']))
plt.scatter(df_Virginica['petal.length'],np.zeros_like(df_Virginica['petal.length']))
plt.scatter(df_Versicolor['petal.length'],np.zeros_like(df_Versicolor['petal.length'])) plt.xlabel('petal.length')
plt.show()

#bivariate sepal.width vs petal.width
sns.FacetGrid(df,hue='variety',height=5).map(plt.scatter,"sepal.width","petal.width").add_legend();
plt.show()

#bivariate sepal.length vs petal.length
sns.FacetGrid(df,hue='variety',height=5).map(plt.scatter,"sepal.length","petal.length").add_legend();
plt.show()

#multivariate all the features
sns.pairplot(df,hue="variety",size=2)
```



Univariate Analysis: Sepal Width

Univariate Analysis: Petal Lengt

Univariate Analysis: Petal Width

Univariate Analysis: Petal Leng

Bivariate Analysis: Sepal Width vs Petal Width

Bivariate Analysis: Sepal Lengt vs Petal Length

Multivariate Analysis: Pairpiot of All Features

RESULT:

Thus a python program to implement univariate, bivariate and multivariate regression features for a given iris dataset is written and the output ius verified.