Ex No: 1

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 Linear Regression 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.

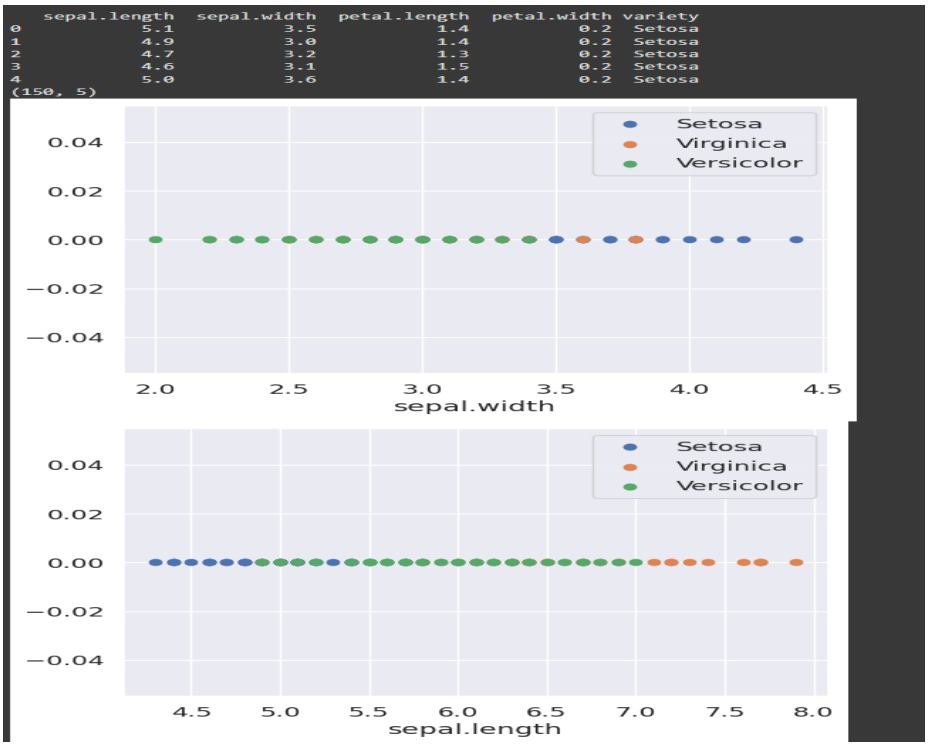
● Run the program to perform univariate, bivariate, and multivariate regression on the dataset.

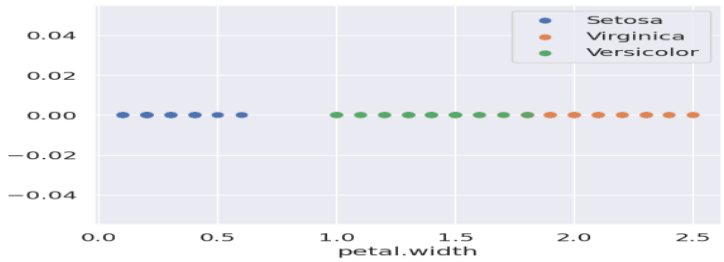
PROGRAM:

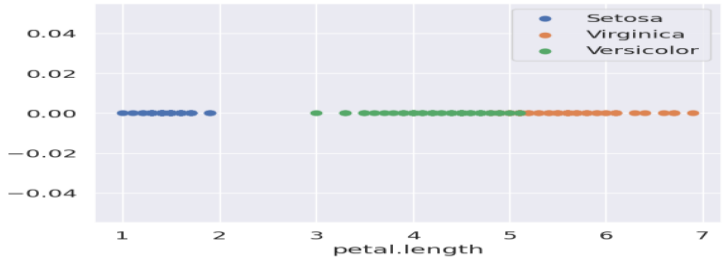
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
df = pd.read_csv('/content/iris - iris.csv')
print(df.head())
print(df.shape)
df_Setosa = df[df['variety'] == 'Setosa']
df_Virginica = df[df['variety'] == 'Virginica']
df_Versicolor = df[df['variety'] == 'Versicolor']
plt.scatter(df_Setosa['sepal.width'],
np.zeros_like(df_Setosa['sepal.width']), label='Setosa')
plt.scatter(df_Virginica['sepal.width'],
np.zeros_like(df_Virginica['sepal.width']), label='Virginica')
plt.scatter(df_Versicolor['sepal.width'],
np.zeros_like(df_Versicolor['sepal.width']), label='Versicolor')
plt.xlabel('sepal.width')
plt.legend()
plt.show()
plt.scatter(df_Setosa['sepal.length'],
np.zeros_like(df_Setosa['sepal.length']), label='Setosa')
plt.scatter(df_Virginica['sepal.length'],
np.zeros_like(df_Virginica['sepal.length']), label='Virginica')
plt.scatter(df_Versicolor['sepal.length'],
np.zeros_like(df_Versicolor['sepal.length']), label='Versicolor')
plt.xlabel('sepal.length')
plt.legend()
plt.show()
plt.scatter(df_Setosa['petal.width'],
np.zeros_like(df_Setosa['petal.width']), label='Setosa')
plt.scatter(df_Virginica['petal.width'],
np.zeros_like(df_Virginica['petal.width']), label='Virginica')
plt.scatter(df_Versicolor['petal.width'],
np.zeros_like(df_Versicolor['petal.width']), label='Versicolor')
plt.xlabel('petal.width')
plt.legend()
plt.show()
plt.scatter(df_Setosa['petal.length'],
np.zeros_like(df_Setosa['petal.length']), label='Setosa')
plt.scatter(df_Virginica['petal.length'],
np.zeros_like(df_Virginica['petal.length']), label='Virginica')
plt.scatter(df_Versicolor['petal.length'],
np.zeros_like(df_Versicolor['petal.length']), label='Versicolor')
```

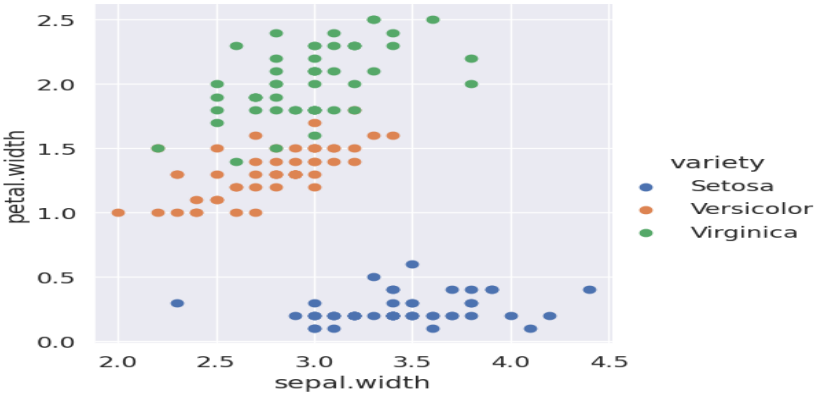
```
plt.xlabel('petal.length')
plt.legend()
plt.show()
sns.FacetGrid(df, hue='variety', height=5).map(plt.scatter, "sepal.width",
"petal.width").add_legend()
plt.show()
sns.FacetGrid(df, hue='variety', height=5).map(plt.scatter,
"sepal.length", "petal.length").add_legend()
plt.show()
sns.pairplot(df, hue="variety", height=2)
plt.show()
```

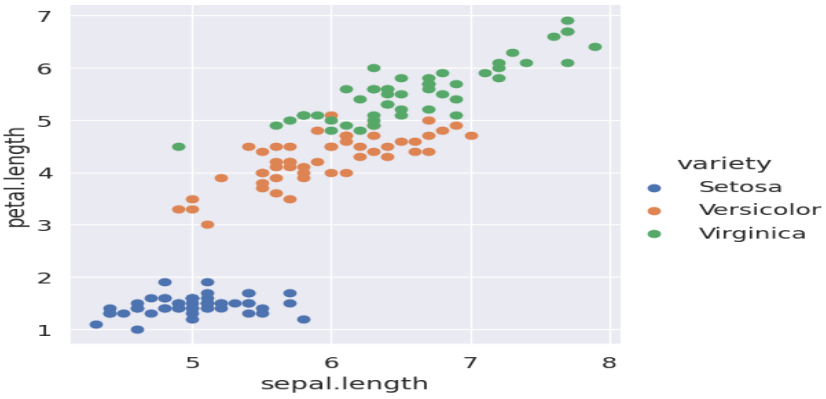
OUTPUT:

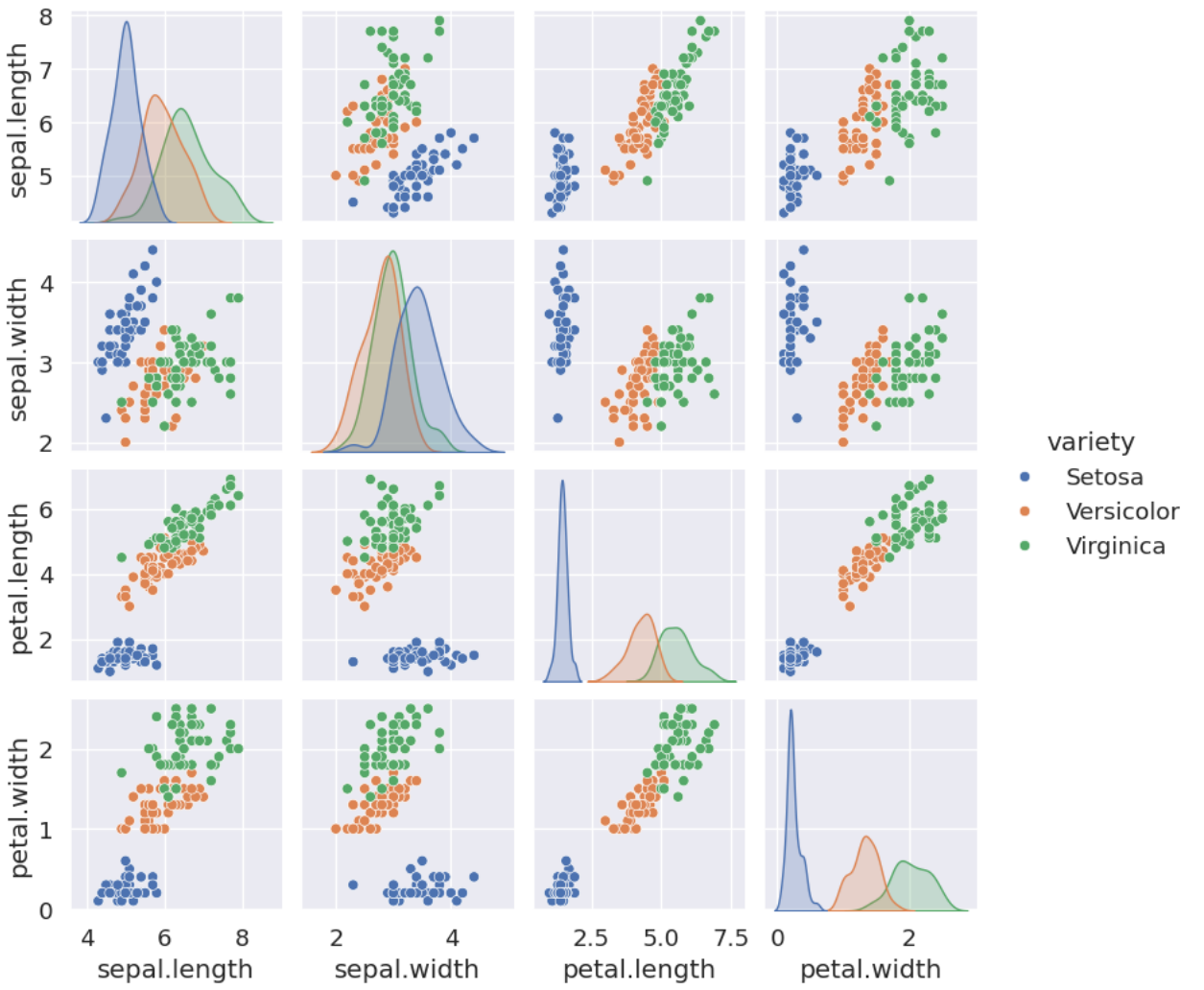



Setosa
Virginica
Versicolor
0.04
0.02
0.00
−0.02
−0.04
0.0
0.5
1.0
1.5
2.0
2.5
petal.width


Setosa
Virginica
Versicolor
0.04
0.02
0.00
−0.02
−0.04
1
2
3
4
5
6
7
petal.length


2.5
2.0
1.5
1.0
0.5
0.0
petal.width
2.0
2.5
3.0
3.5
4.0
4.5
sepal.width
variety
Setosa
Versicolor
Virginica


7
6
5
4
3
2
1
petal.length
5
6
7
8
sepal.length
variety
Setosa
Versicolor
Virginica



RESULT: -

Thus, the Python program to implement univariate, bivariate, and multivariate regression features for the given dataset is analyzed, and the features are plotted using a scatter plot.