**Objective**:

To write a python program that uses the Iris dataset provided in the scikit-learn package to train a Linear Regression model from scratch using gradient descent to properly predict the class of an instance of an Iris flower and to plot the data actual vs predicted using the matplotlib graphics library.

**Procedure:**

1. Use Spyder IDE to write the python program.

2. Load the Irisdata using Scikit Learn (Alternatively I used numpy library genfromtxt function to read the dataset downloaded from the official website URL: “https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data”).

3. Sample the dataset into training and testing samples by using the train\_test\_split function that should be imported from the scikit learn cross\_validation class.

4. Give the test\_size parameter as 0.2 and random\_state as any number (3 in my case).

5. Train the linear regression model by passing the training sample set as parameters.

6. Predict the result by passing the test data sample

7. Using the Matplotlib graphics library plot the actual sample set Vs predicted output

**Technology used:** Python. Libraries Used are numpy, sklearn, matplotlib

**Tool used:** Spyder

**Source Code Filename:**

MyLinearRegressionModel.py

**Program Explanation:**

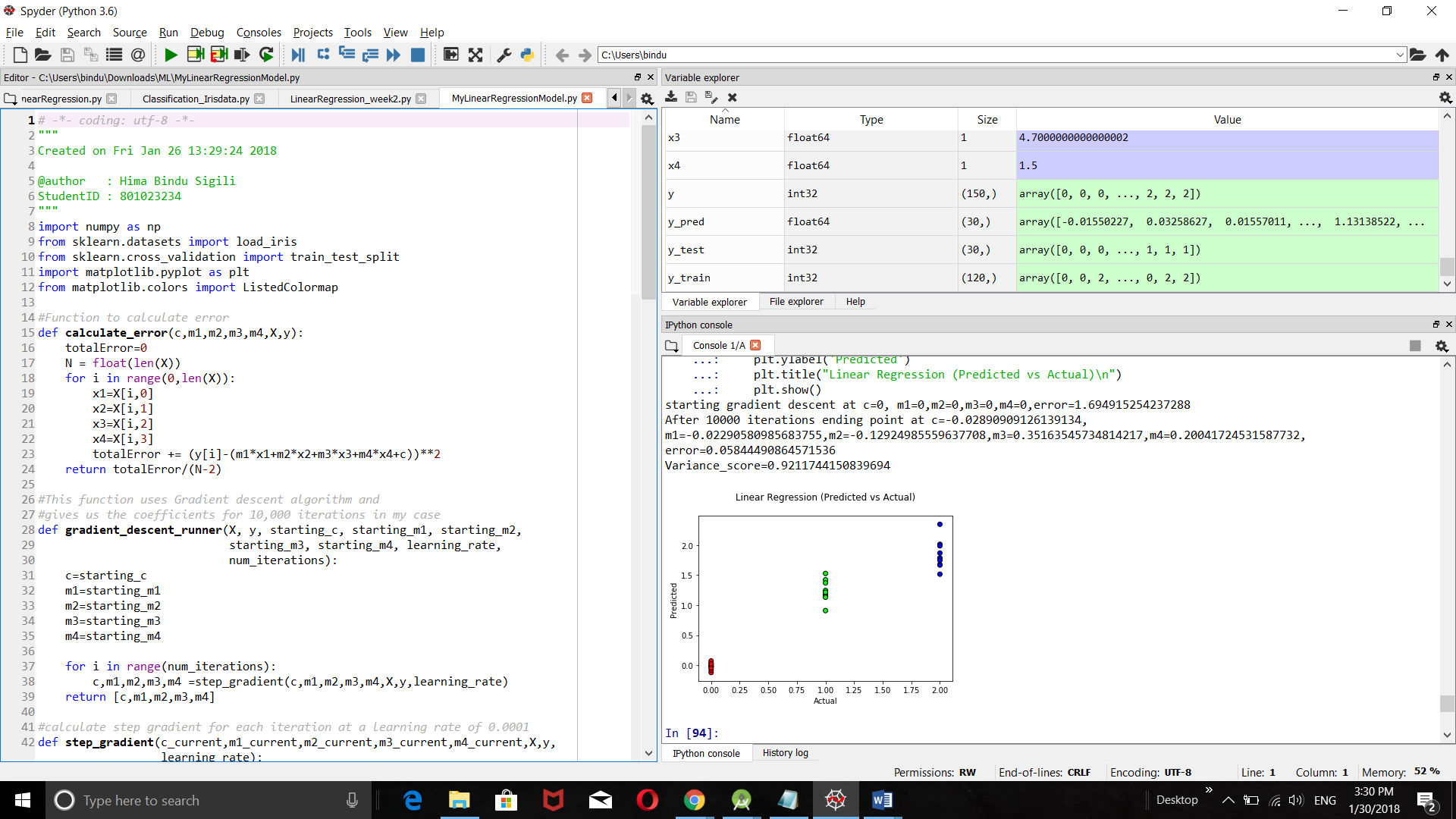
1. The program is to load the Iris data, build our own model, predict the output and plot the output actual vs predicted using scikit learn and matplotlib packages.
2. Use the Iris dataset from the Scikit learn sample datasets.
3. Or else use numpy genfromtxt function, the program takes csv file as input (the Iris data that is downloaded from the official website). Link to download: <https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data> Import this from the excel and save it as .csv file(code for this alternative is commented in the program for reference).
4. Use scikitlearn train\_test\_split function to split the data into training and testing samples.(Also we can split using shuffle and slice functions but random shuffle results in different samples every time so I used train\_test\_split function, the alternate code is commented for reference)
5. The main function calls the training function where it internally calls calculate\_ error method to calculate the error before and after the training and gradient\_descent method to train the data.
6. We can obtain the coefficients for our linear equation (Cost function)
7. Using this equation, we predict the output for the testing sample data.
8. Got coefficients as m1(for sepal length) =-0.02, m2(sepal width) =-0.12, m3(petal length) =0.35,

m4(petal width) =0.2 and c(bias)= -0.03. Mean squared error = 0.05 and Variance score = 0.92

1. To calculate variance score referred GitHub code of the official scikit learn website at <https://github.com/scikit-learn/scikit-learn/blob/a24c8b46/sklearn/metrics/regression.py#L444>
2. Finally, Actual (test data) Vs Predicted data is plotted.

**Screen Shots of output:**

1.Where x and y-axis are Actual Data and Predicted Data respectively.



**Conclusion:**

Linear Regression model has been designed with an MSE of 0.05 and Variance score of 0.92. Got familiar with the python language, libraries, understood the concept and implemented linear regression model.