**QUESTION 1:**

Using the attached dataset to develop, train, and evaluate a group of linear regression models to predict the price (dependent variable) of a Monet painting from a few of its features (independent variables). Create your model in Python.

Tasks: 1. Create at least two simple linear regression models, each of them has one different independent variable (you may transform the raw independent variable into different formats, such as to conduct a logarithmic transformation or combine two variables into a new variable such as Size = width \* height). You may consider one variable as Size, and another one as Width. Create a scatter plot for showing the relationship between the independent variable and the dependent variable for each model, and showing the linear regression line in the same plot. Calculate the error of the prediction with test data.

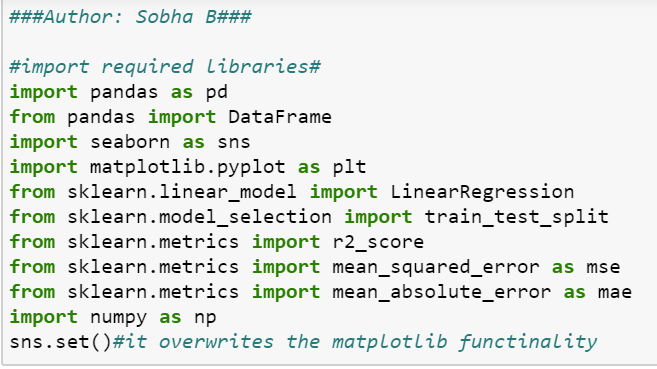
**SOLUTION:**

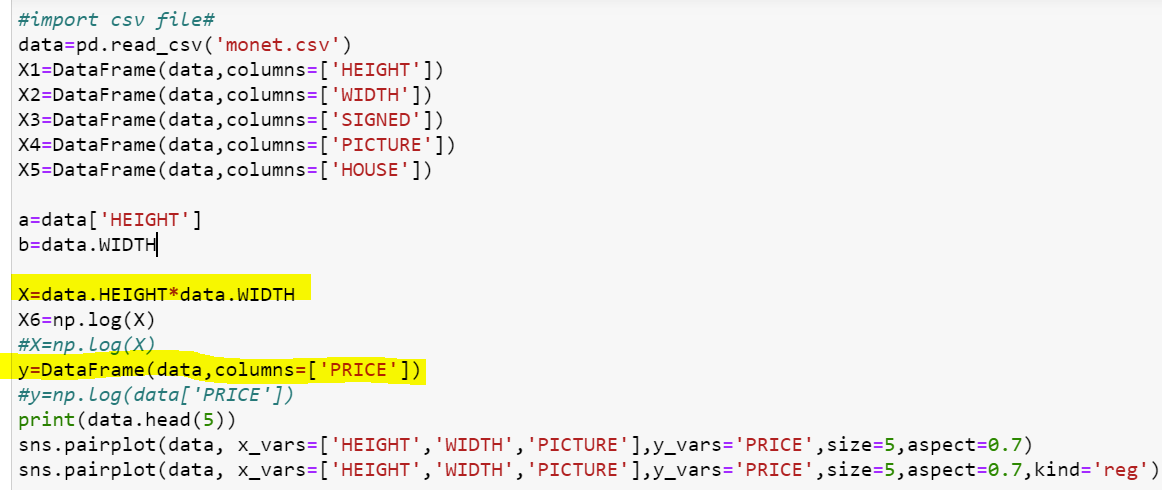
Attached is the Python code generated in Jupyter NB.

**TASK 1:**

1. **SIMPLE LINEAR REGRESSION (SLR\_Size.ipynb):**

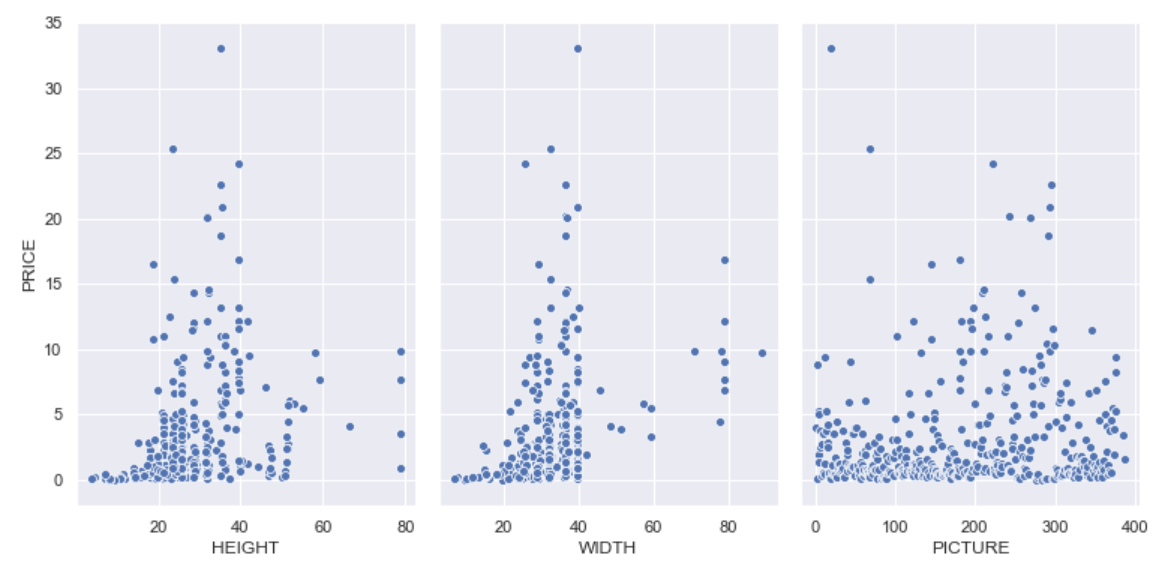
* As per the task requirement, I have imported the required libraries and read the monet.CSV file. From which, I have read all the variables.
* I have created a new variable called “Size” by multiplying HEIGHT and WIDTH.
* For this simple linear regression, I am going to use **SIZE** as the **independent** variable and **PRICE** as the **dependent** variable.
  + - * 1. **Scatter Plots:**
* To explore how the data has been distributed, I have created a scatter plots for “HEIGHT”, “WIDTH”, “PICTURE” using seaborn and “SIZE” variables .



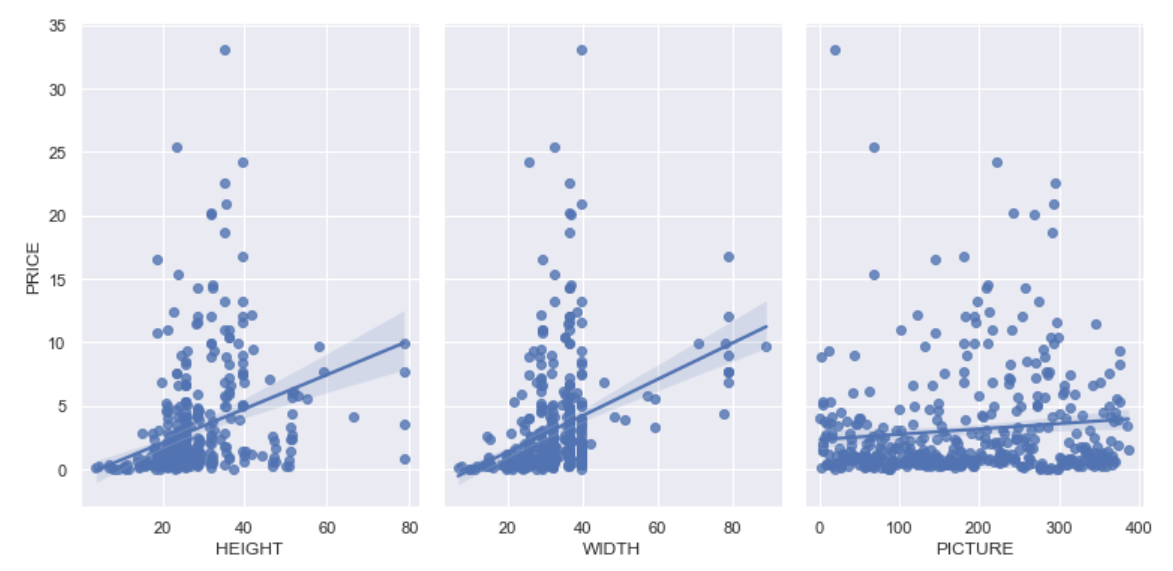


**Output:**

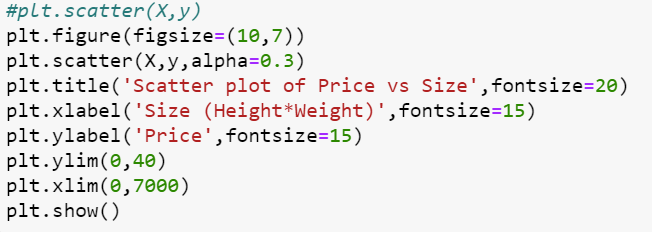
**Scatter plots of respective variables.**

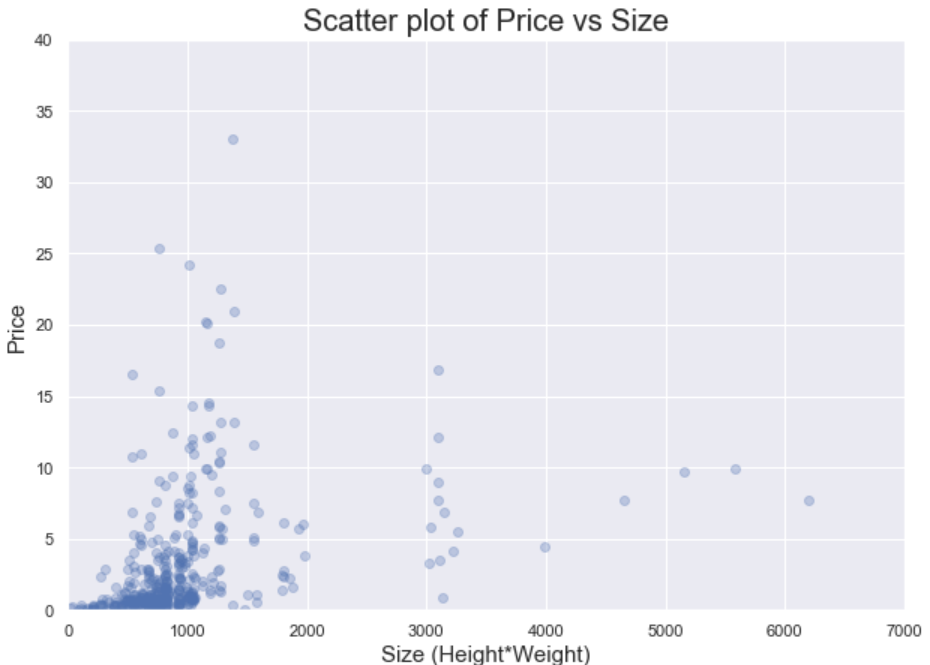


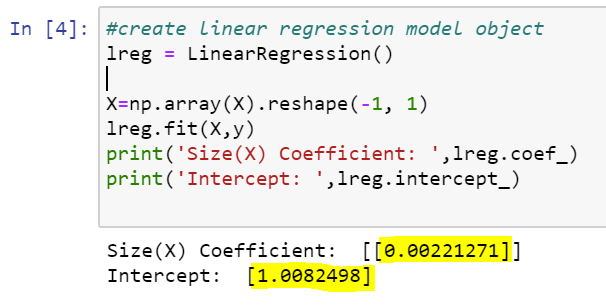
**Scatter plots of respective variables with regression line.**



* Below is the scatter plot for Size and Price.



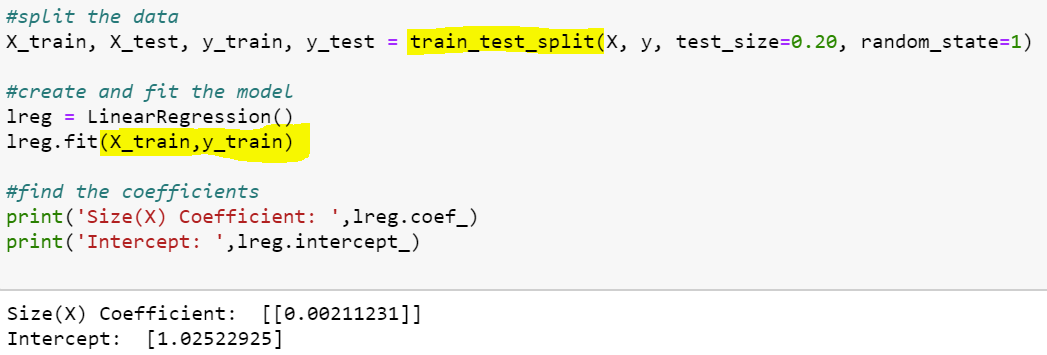


* + - * 1. **Develop a Model:**
* Once the dependent and independent variables are identified and drawn the scatter plots. The next step is to develop a model.
* I have used Sklearn to develop, train and evaluate linear regression method. From the model, I have identified the co-efficient and intercept.
* With the help of these Intercept and Coefficient, now we can draw the regression line using below code.

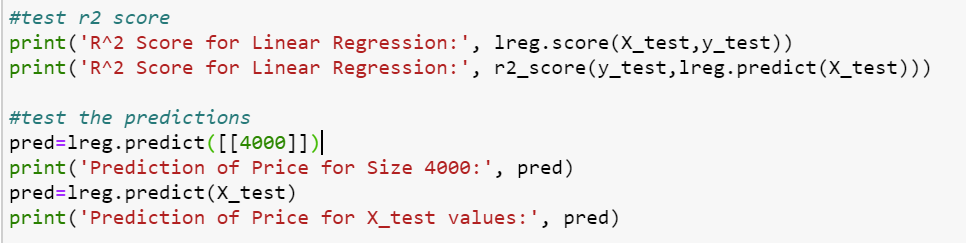




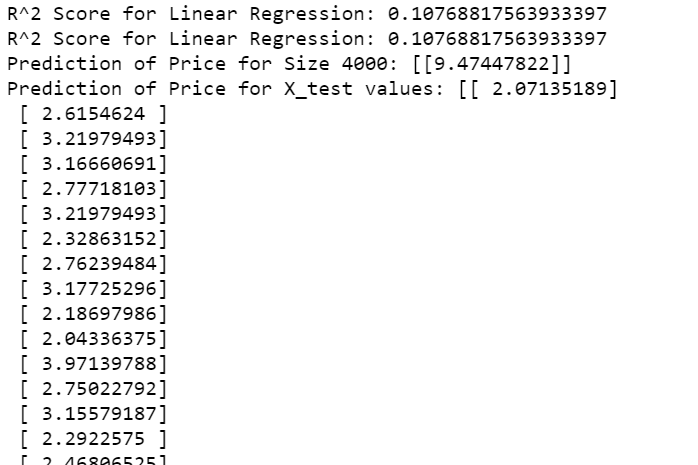
* + - * 1. **Train the data:**
* The data has been trained, and splitted in the ration of 80:20 for train and test respectively.
* Now the co-efficient and intercept were calculated based on the split data.
* It is straight forward to train regression and make predictions with scikit learn.
* As a first step, I have used the “LinearRegression()” function from linear\_model module.
* Once the model has selected, I have used fit() function to fit the model on the train set and subsequently used predict() function to perform prediction on the test data.



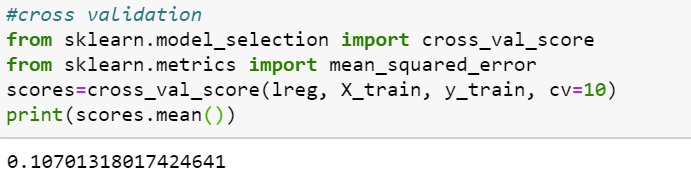
* + - * 1. **Calculation of R-square and the Predictions:**
* Now, I have calculated the **R-square** to know the goodness of fit and finally **predicted** the results.
* I have checked the R^2 score using the “lreg.score” function on the test data.
* R^2 value should be 0 to 1, which indicates that the data is nearly linear and the independent variable (like size) is correlated to the price.



**Output:**

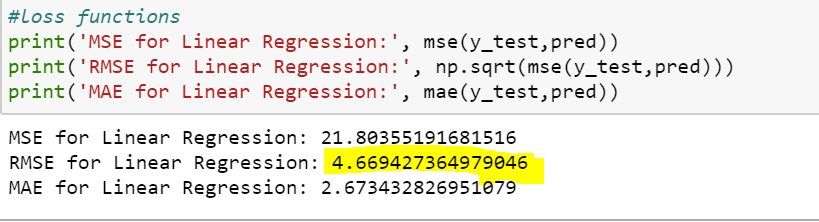


* + - * 1. Cross-validation using k-fold cross validation method:
* In this step, I am going to create a subset of training set, called validation set using a cross\_val\_score() function.
* K-fold cross validation involves randomly dividing the training set into k different folds/ groups of approximately equal size. The first fold is treated as a validation set and the method is fit on the remaining k-1 folds.
* I have used a fold of 10 (cv=10). Hence, 10 folds will be created for cross\_val\_score function.
* For each fold, one value will be generated, however we should take the mean of 10 folds. The obtained score value is almost near to the R^2 score and hence it is validated.



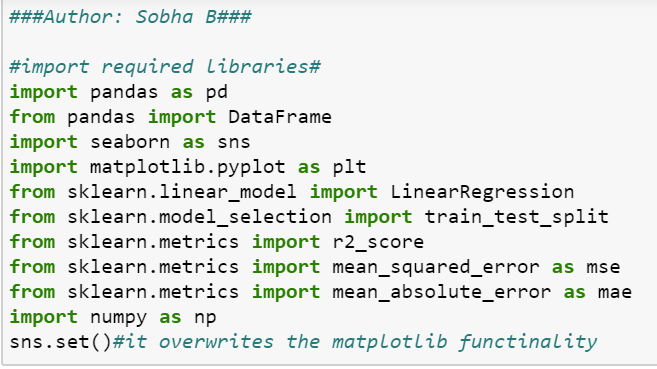
**(f) Loss Functions/ Errors of Prediction:**

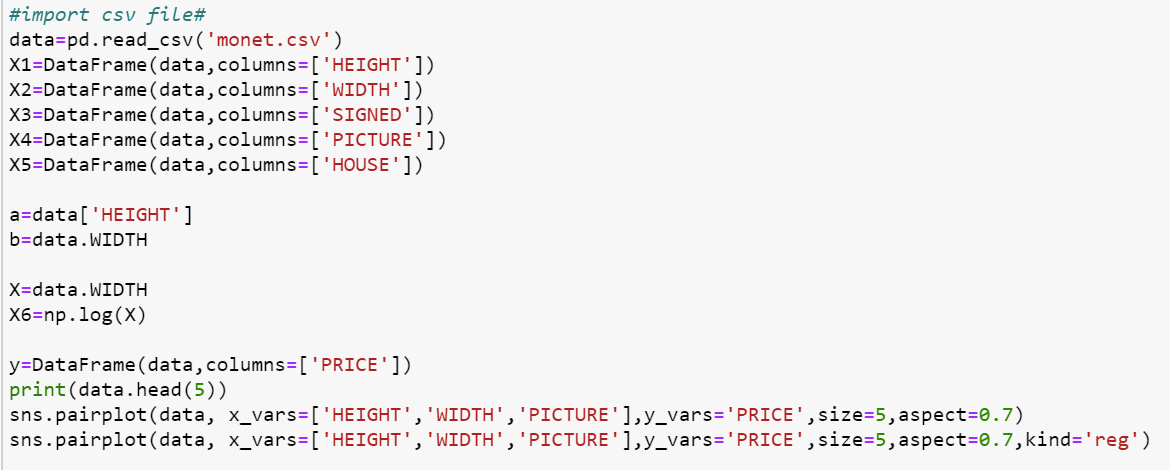
* In regression functions, where quantity is predicted, we use mean squared error (MSE) loss function.
* Few basic functions are commonly used, out of which MSE is the popular regression problems.
* Mean squared error (MSE), measures the [average](https://en.wikipedia.org/wiki/Expected_value) of the squares of the [errors](https://en.wikipedia.org/wiki/Error_(statistics)), i.e., that is, the average squared difference between the estimated values and the actual values.
* RMSE, MAE can also be considered.
* Mean absolute error (MAE) is a measure of [errors](https://en.wikipedia.org/wiki/Error_(statistics)) between paired observations expressing the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed values.
* Whereas, RMSE is the square root of MSE
* These can be calculated by importing modules like mean\_absolute\_error, mean\_squared\_error from metrics.
* I ideal values of loss functions are 0, lower the loss, the good is the data. However, there are no hard definitions to define the ranges.



1. **SIMPLE LINEAR REGRESSION (SLR\_width.ipynb):**

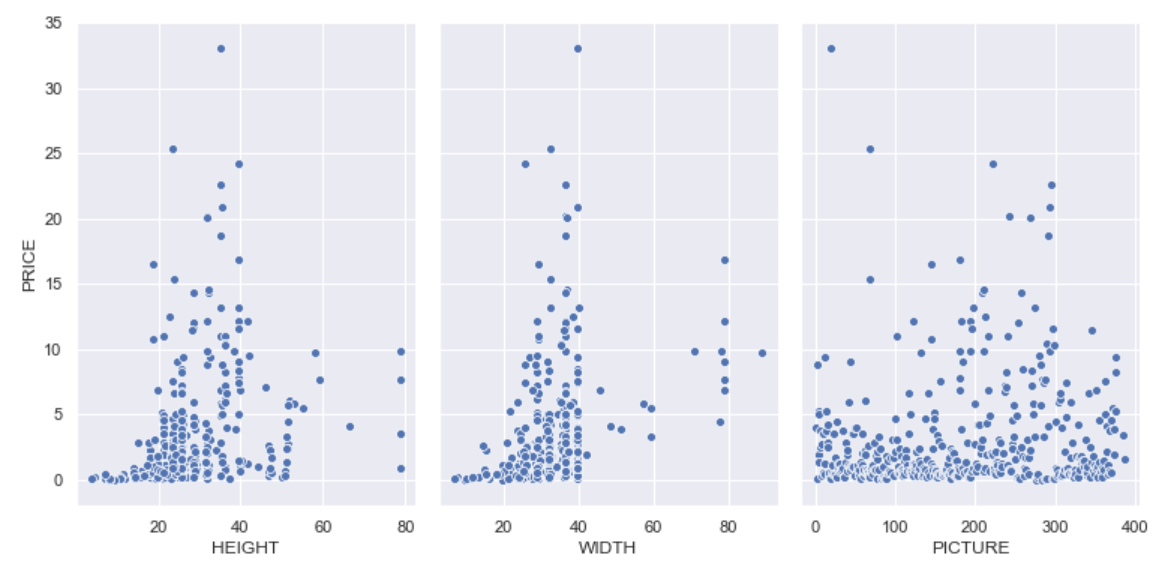
* For this simple linear regression, I am going to use **WIDTH** as the **independent** variable and **PRICE** as the **dependent** variable.
  + - * 1. **Scatter Plots:**
* To explore how the data has been distributed, I have created a scatter plots for “HEIGHT”, “WIDTH”, “PICTURE” using seaborn.



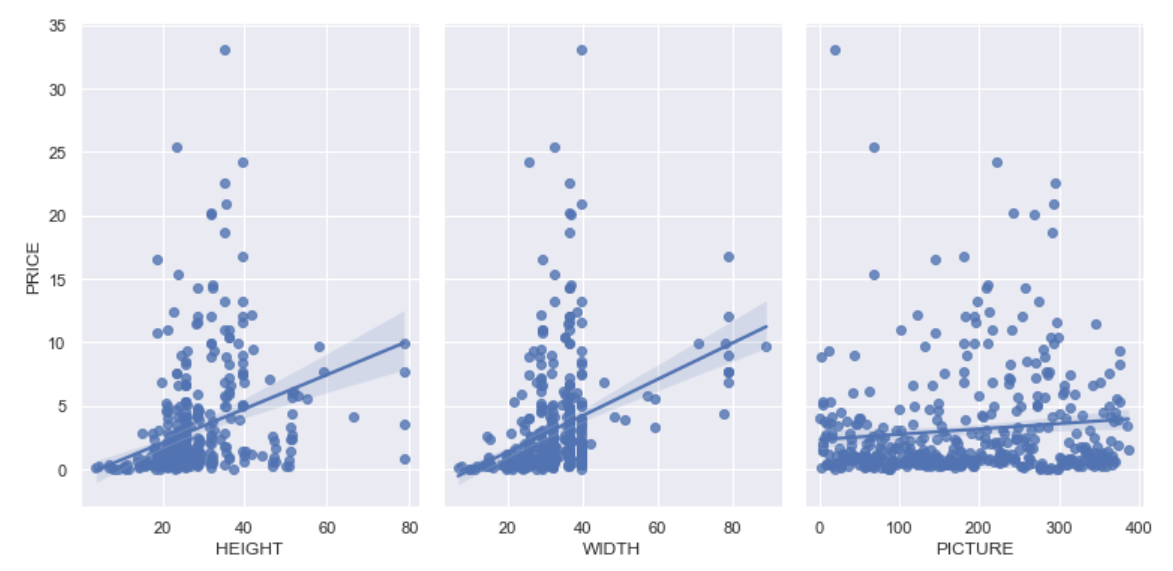


**Output:**

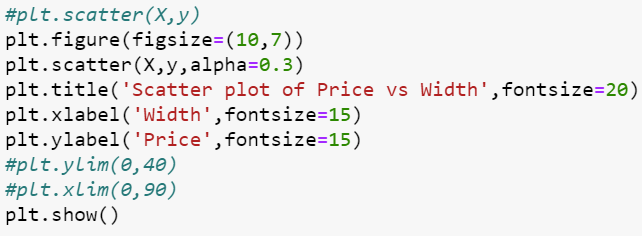
**Scatter plots of respective variables.**

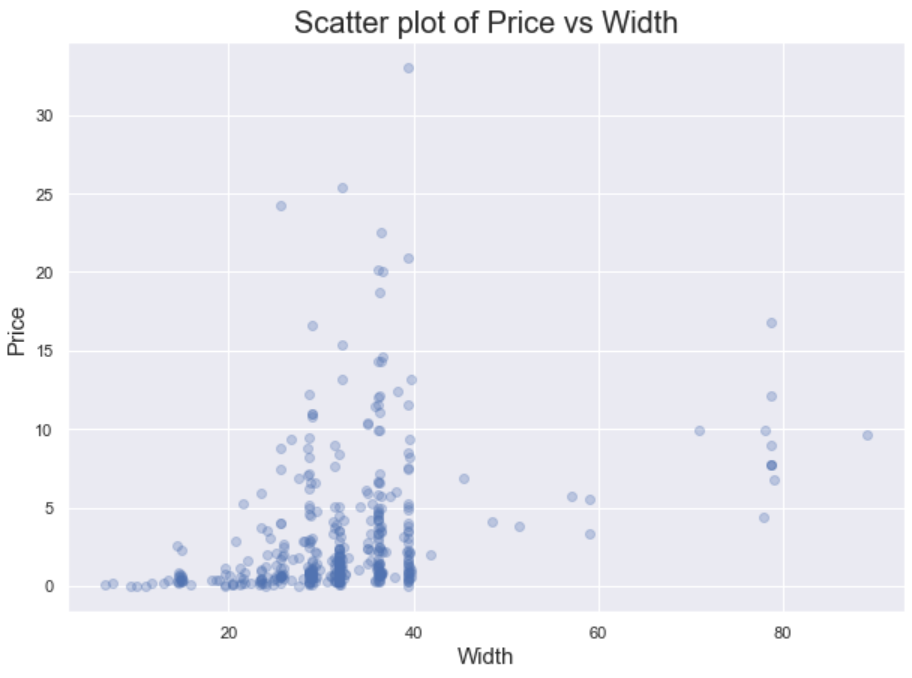


**Scatter plots of respective variables with regression line.**

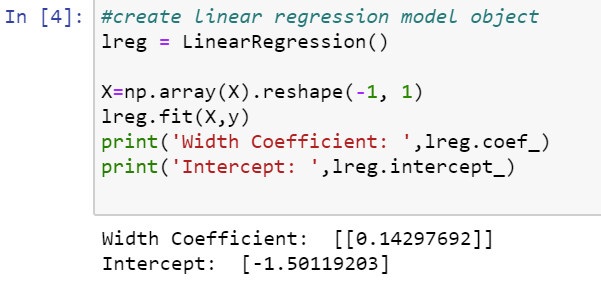


* Below is the scatter plot for width and Price.





* + - * 1. **Develop a Model:**
* Once the dependent and independent variables are identified and drawn the scatter plots. The next step is to develop a model.
* I have used Sklearn to develop, train and evaluate linear regression method. From the model, I have identified the co-efficient and intercept.

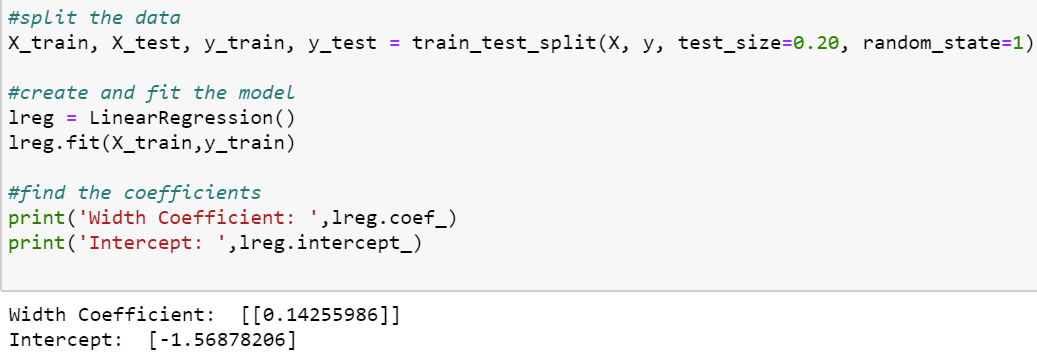


* With the help of these Intercept and Coefficient, now we can draw the regression line using below code.

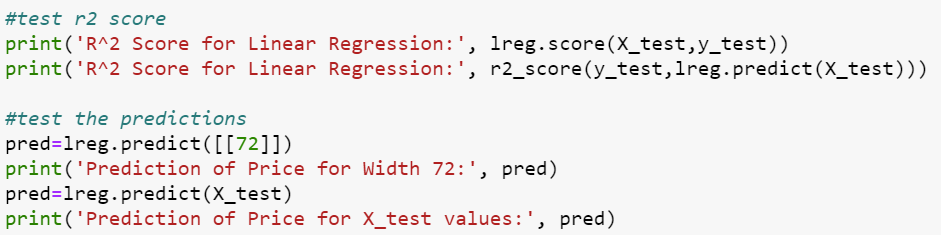




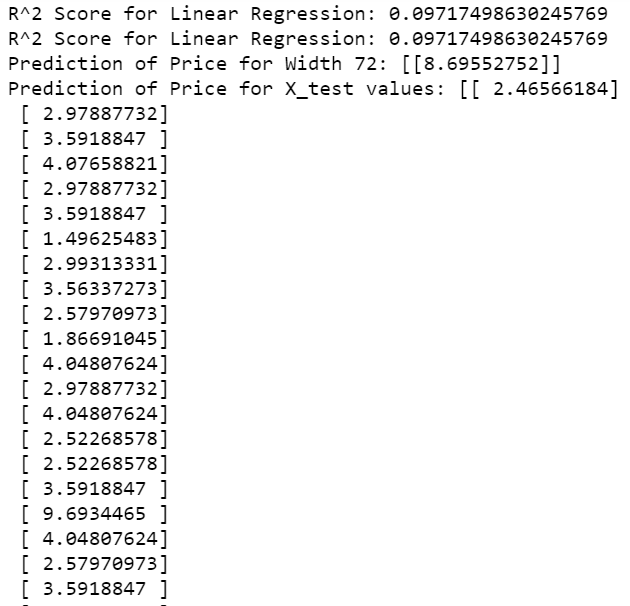
* + - * 1. **Train the data:**
* The data has been trained, and splitted in the ration of 80:20 for train and test respectively.
* Now the co-efficient and intercept were calculated based on the split data.
* It is straight forward to train regression and make predictions with scikit learn.
* As a first step, I have used the “LinearRegression()” function from linear\_model module.
* Once the model has selected, I have used fit() function to fit the model on the train set and subsequently used predict() function to perform prediction on the test data.



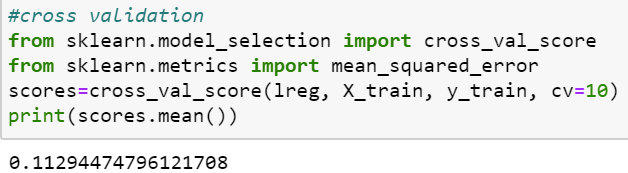
* + - * 1. **Calculation of R-square and the Predictions:**
* Now, I have calculated the **R-square** to know the goodness of fit and finally **predicted** the results.
* I have checked the R^2 score using the “lreg.score” function on the test data.
* R^2 value should be 0 to 1, which indicates that the data is nearly linear and the independent variable (like size) is correlated to the price.



**Output:**

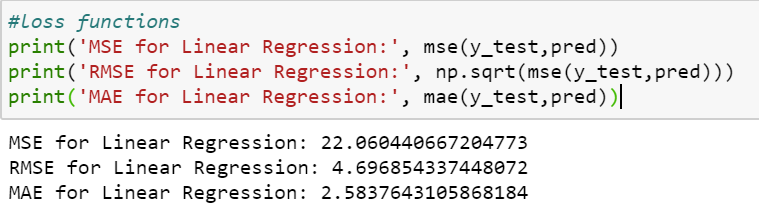


* + - * 1. Cross-validation using k-fold cross validation method:
* In this step, I am going to create a subset of training set, called validation set using a cross\_val\_score() function.
* K-fold cross validation involves randomly dividing the training set into k different folds/ groups of approximately equal size. The first fold is treated as a validation set and the method is fit on the remaining k-1 folds.
* I have used a fold of 10 (cv=10). Hence, 10 folds will be created for cross\_val\_score function.
* For each fold, one value will be generated, however we should take the mean of 10 folds. The obtained score value is almost near to the R^2 score and hence it is validated.



**(f) Loss Functions/ Errors of Prediction:**

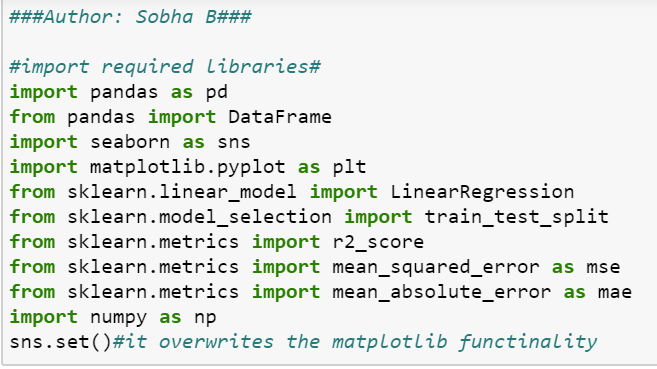
* In regression functions, where quantity is predicted, we use mean squared error (MSE) loss function.
* Few basic functions are commonly used, out of which MSE is the popular regression problems.
* Mean squared error (MSE), measures the [average](https://en.wikipedia.org/wiki/Expected_value) of the squares of the [errors](https://en.wikipedia.org/wiki/Error_(statistics)), i.e., that is, the average squared difference between the estimated values and the actual values.
* RMSE, MAE can also be considered.
* Mean absolute error (MAE) is a measure of [errors](https://en.wikipedia.org/wiki/Error_(statistics)) between paired observations expressing the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed values.
* Whereas, RMSE is the square root of MSE
* These can be calculated by importing modules like mean\_absolute\_error, mean\_squared\_error from metrics.
* I ideal values of loss functions are 0, lower the loss, the good is the data. However, there are no hard definitions to define the ranges.

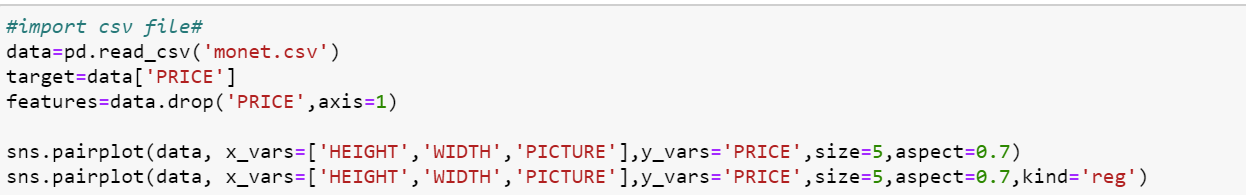


**TASK 2:**

**MULTIVARIABLE REGRESSION (MLR (2 vars).ipynb/ MLR (5 vars)):**

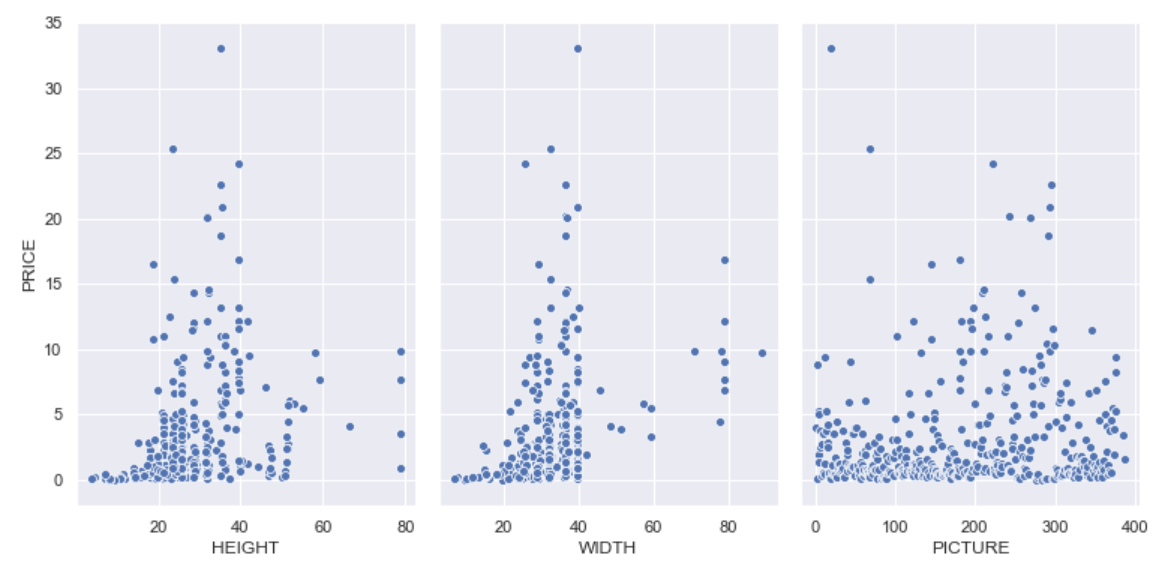
* For this Multi linear regression, I am going to use 2/ 5 **independent** variable and **PRICE** as the **dependent** variable. Below is an example for 2 variables regression.
* For, multi(5)variable regression see the code. Logarithmic values are not considered as price values are going in negative.
  + - * 1. **Scatter Plots:**
* To explore how the data has been distributed, I have created a scatter plots for “HEIGHT”, “WIDTH”, “PICTURE” using seaborn.



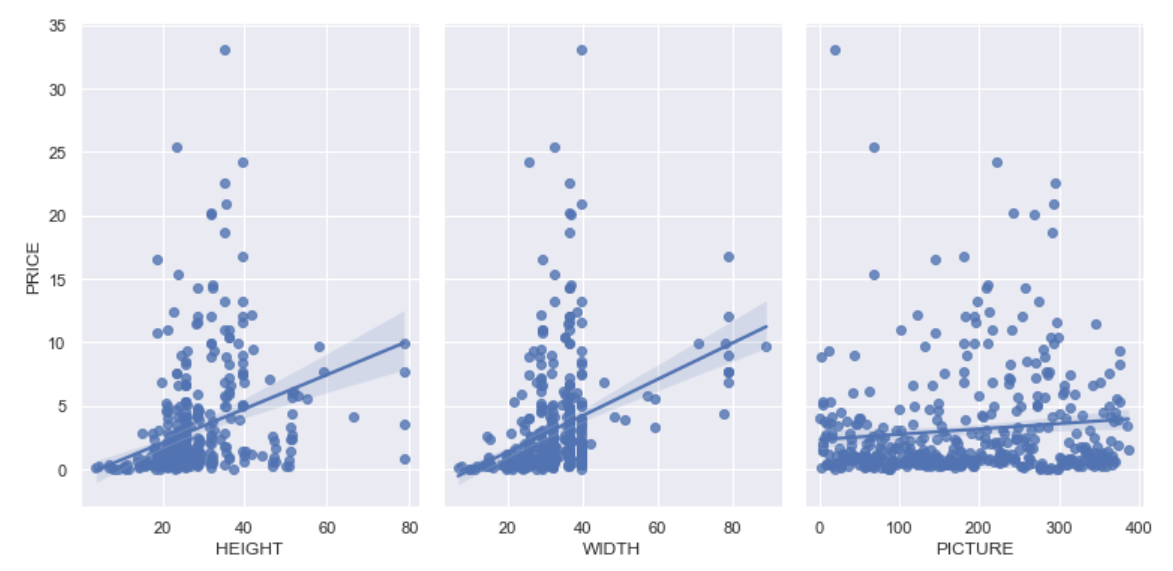


**Output:**

**Scatter plots of respective variables.**

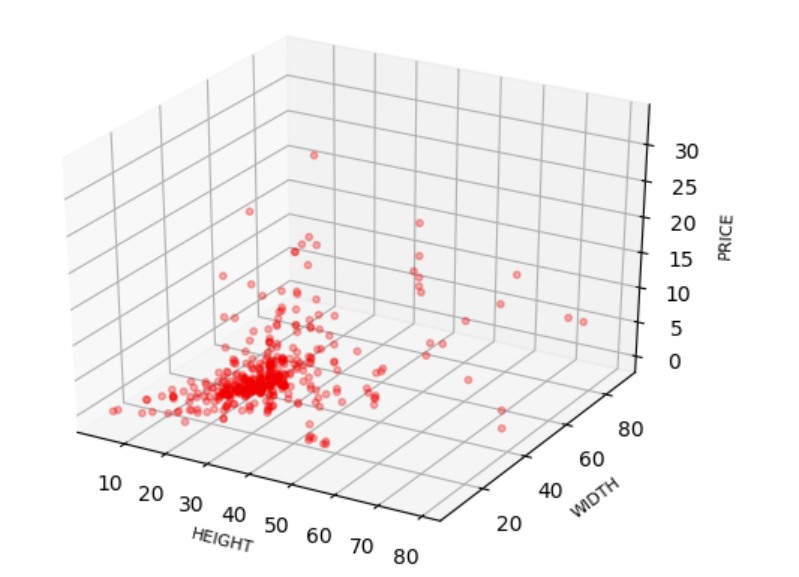


**Scatter plots of respective variables with regression line.**

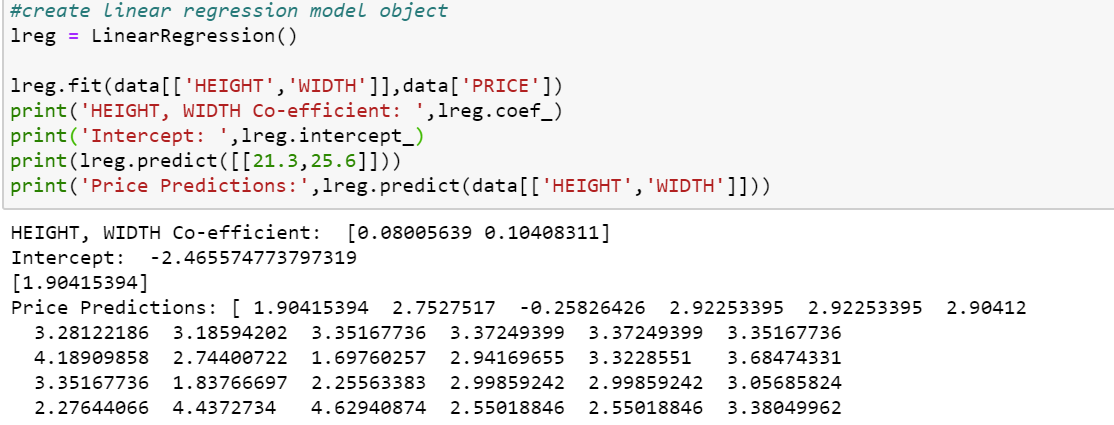


* Below is the 3D scatter plot for width and Price.



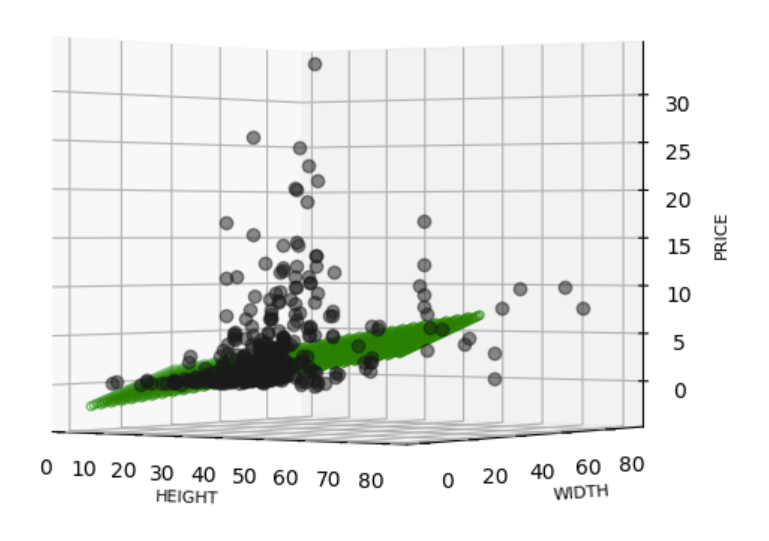


* + - * 1. **Develop a Model:**
* Once the dependent and independent variables are identified and drawn the scatter plots. The next step is to develop a model.
* I have used Sklearn to develop, train and evaluate multi linear regression method. From the model, I have identified the co-efficient and intercept.

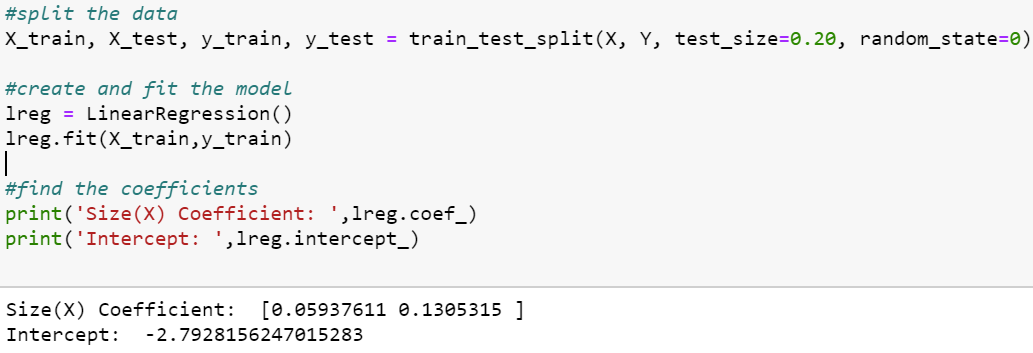


* With the help of these Intercept and Coefficient, now we can draw the regression line using below code.

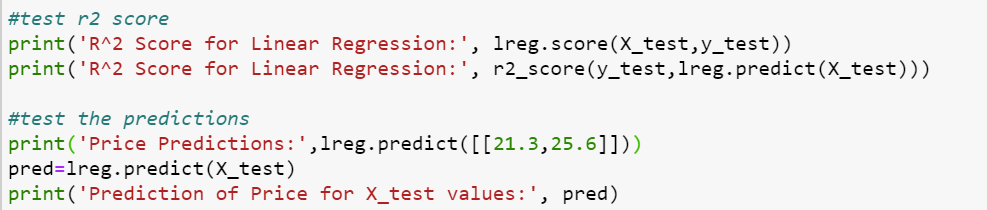




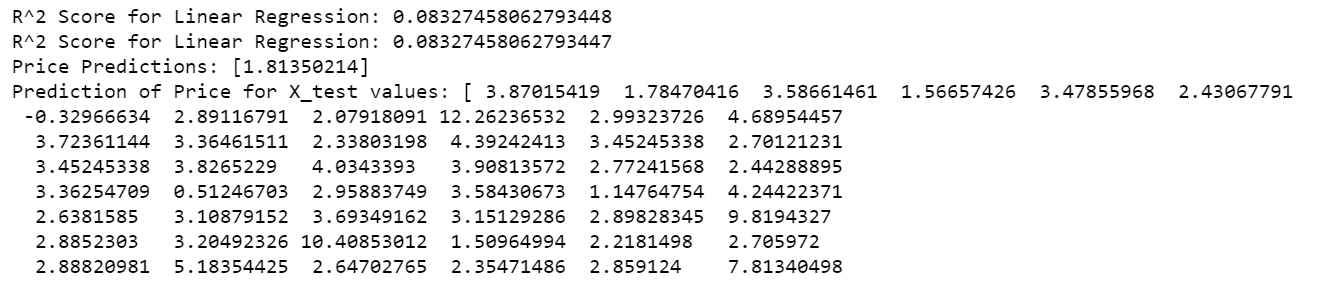
* + - * 1. **Train the data:**
* The data has been trained, and splitted in the ration of 80:20 for train and test respectively.
* Now the co-efficient and intercept were calculated based on the split data.
* It is straight forward to train regression and make predictions with scikit learn.
* As a first step, I have used the “LinearRegression()” function from linear\_model module.
* Once the model has selected, I have used fit() function to fit the model on the train set and subsequently used predict() function to perform prediction on the test data.



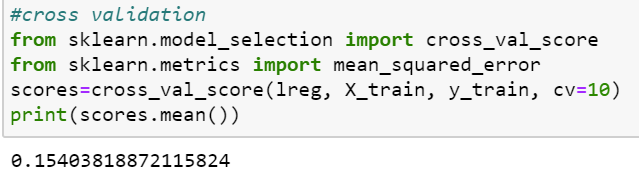
* + - * 1. **Calculation of R-square and the Predictions:**
* Now, I have calculated the **R-square** to know the goodness of fit and finally **predicted** the results.
* I have checked the R^2 score using the “lreg.score” function on the test data.
* R^2 value should be 0 to 1, which indicates that the data is nearly linear and the independent variable (like size) is correlated to the price.



**Output:**



* + - * 1. Cross-validation using k-fold cross validation method:
* In this step, I am going to create a subset of training set, called validation set using a cross\_val\_score() function.
* K-fold cross validation involves randomly dividing the training set into k different folds/ groups of approximately equal size. The first fold is treated as a validation set and the method is fit on the remaining k-1 folds.
* I have used a fold of 10 (cv=10). Hence, 10 folds will be created for cross\_val\_score function.
* For each fold, one value will be generated, however we should take the mean of 10 folds. The obtained score value is almost near to the R^2 score and hence it is validated.



**(f) Loss Functions/ Errors of Prediction:**

* In regression functions, where quantity is predicted, we use mean squared error (MSE) loss function.
* Few basic functions are commonly used, out of which MSE is the popular regression problems.
* Mean squared error (MSE), measures the [average](https://en.wikipedia.org/wiki/Expected_value) of the squares of the [errors](https://en.wikipedia.org/wiki/Error_(statistics)), i.e., that is, the average squared difference between the estimated values and the actual values.
* RMSE, MAE can also be considered.
* Mean absolute error (MAE) is a measure of [errors](https://en.wikipedia.org/wiki/Error_(statistics)) between paired observations expressing the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed values.
* Whereas, RMSE is the square root of MSE
* These can be calculated by importing modules like mean\_absolute\_error, mean\_squared\_error from metrics.
* I ideal values of loss functions are 0, lower the loss, the good is the data. However, there are no hard definitions to define the ranges.

