**INTRODUCTION TO ARTIFICIAL INTELLIGENCE COM526**

**WEEK 6 ACTIVITIES**

Note: This activity **does not need** to be included in AE1 portfolio.

**Linear Regression Models**

Linear regression is a statistical approach that models the relationship between input features and output. The input features are called the independent variables, and the output is called a dependent variable. Our goal here is to predict the value of the output based on the input features.

There are two types of linear regression:

1. Simple Linear Regression

2. Multivariable Linear Regression

In simple linear regression, we predict the output/dependent variable based on only one input feature. The simple linear regression (sloe-intercept form) is given by:

Y= mX + C where

Y= output (dependent/target variable)

m = coefficient of input feature

X= input feature (independent variable) on which output is based

C= constant or y-intercept of line

Sometimes, you will also see the equation in its standard form:

aX + bY = c

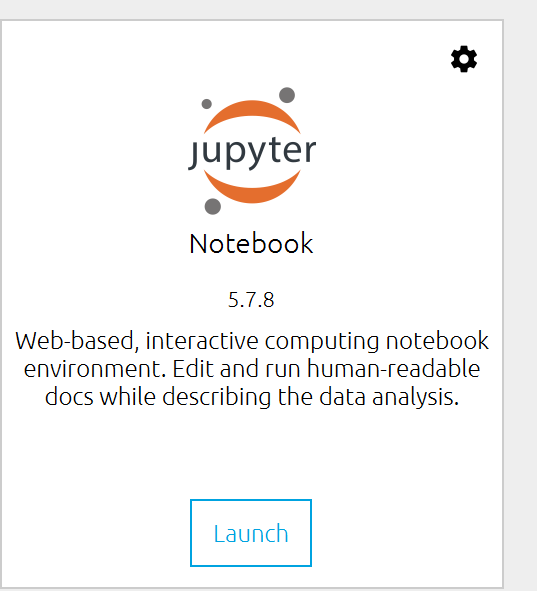
Below we are going to implement simple linear regression using the sklearn library in Python.

**Step by Step Guide**

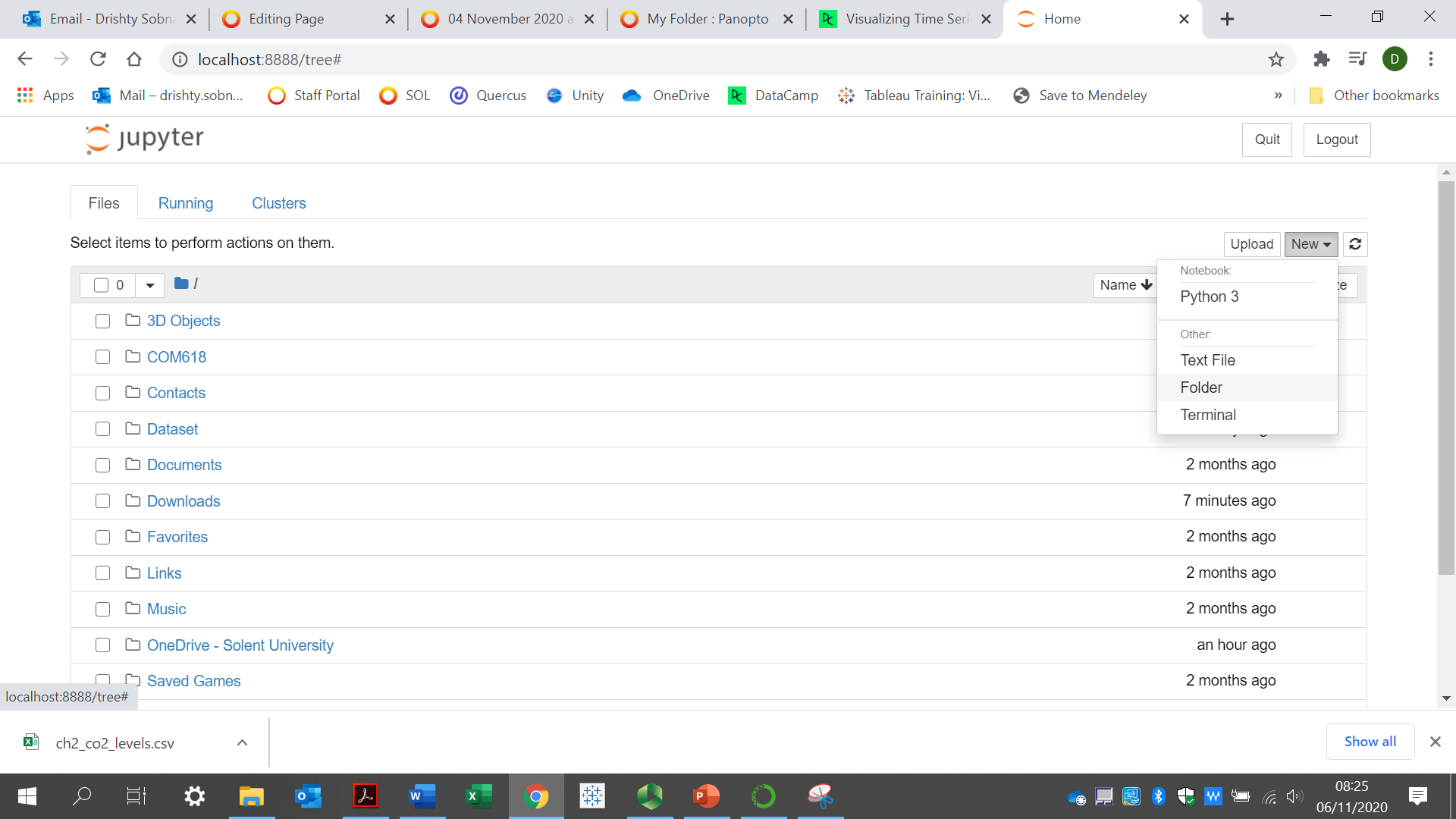
1. First you need to have Anaconda/Jupyter Notebook set up or any alternative platforms to run Python codes. A guide to install Anaconda has been provided to you on SOL.
2. Since we are going to use various libraries for calculations, we need to import them first.

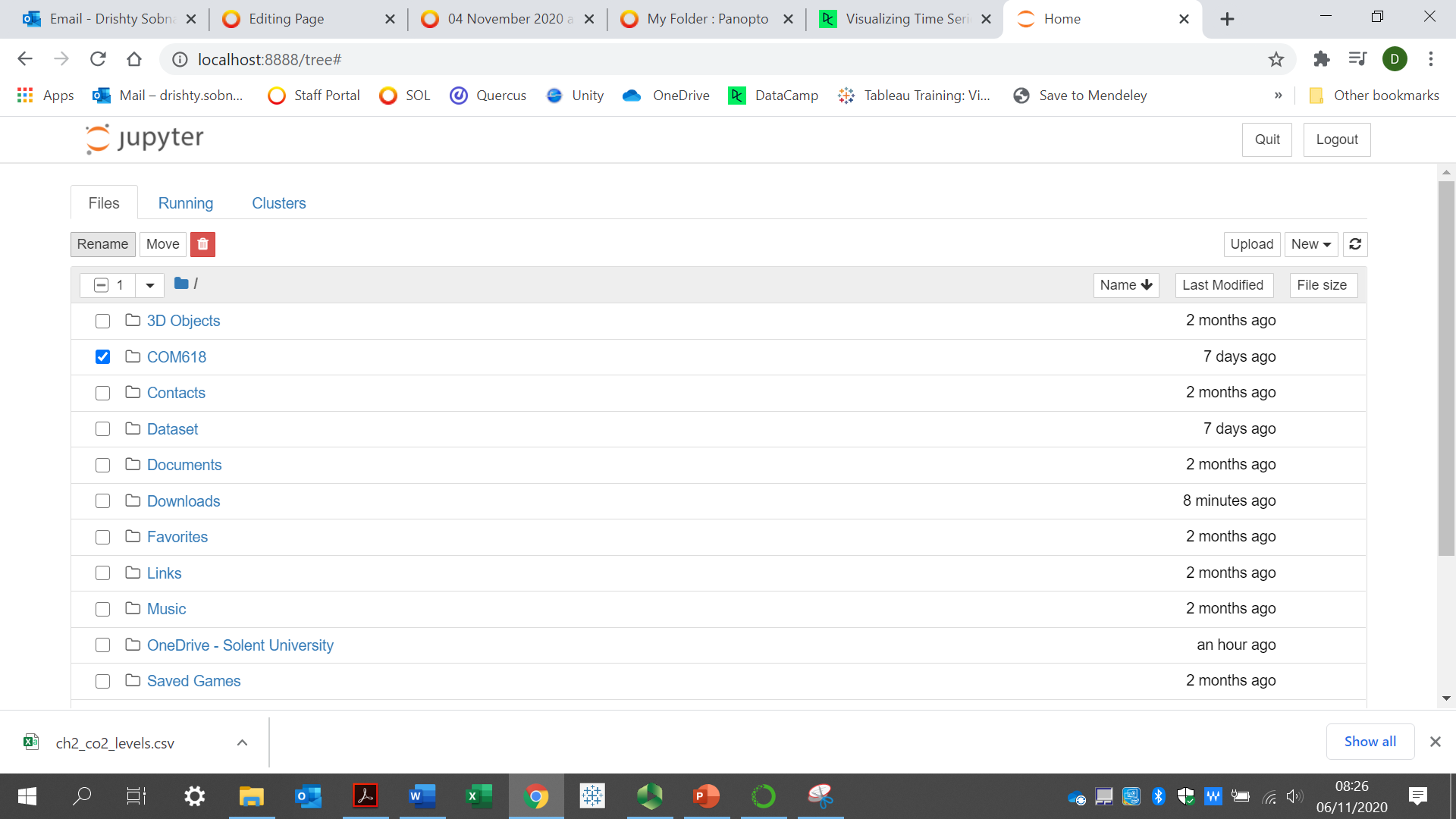
* NumPy: A library that makes a variety of mathematical and statistical operations easier; it is also the basis for many features of the Pandas library.
* Pandas: A Python library created specifically to facilitate working with data, this one of the main libraries Python data science work.
* Matplotlib: A visualisation library that makes it quick and easy to generate charts from your data.
* scikit-learn: The most popular library for machine learning work in Python.

1. Open Anaconda app that you have downloaded and clink on the Launch button to open Jupyter Notebook as follows:

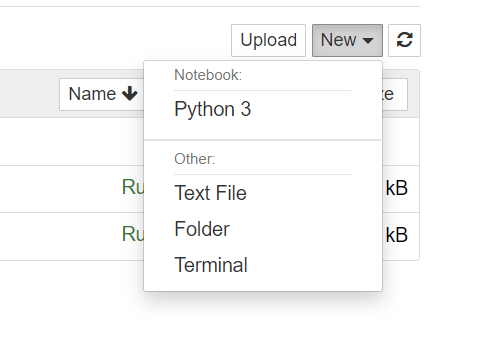


1. Create a folder and name it COM526. To name/rename a folder, select the folder and click on Rename.

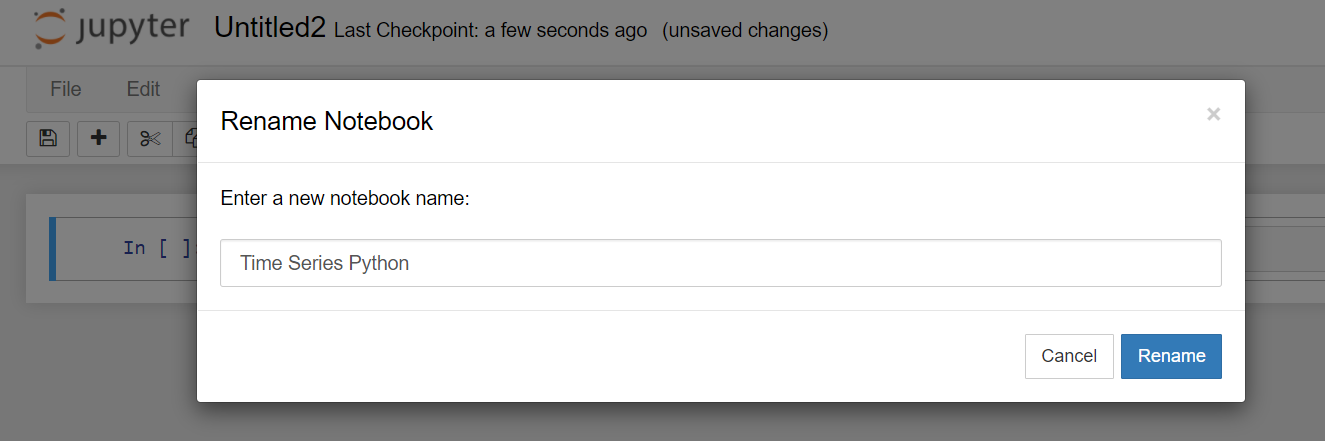




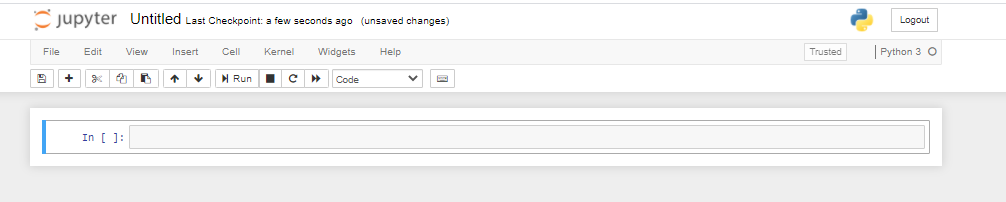
1. Now that you have created your folder, you can create a Python file. Click on New, Python 3.



1. Once your Python file is created, you can rename it by clicking on the “Untitled” file and rename it to Linear Regression.



1. You can now enter your Python codes here.



Type the following codes to import the necessary libraries and **click on the Run button**:



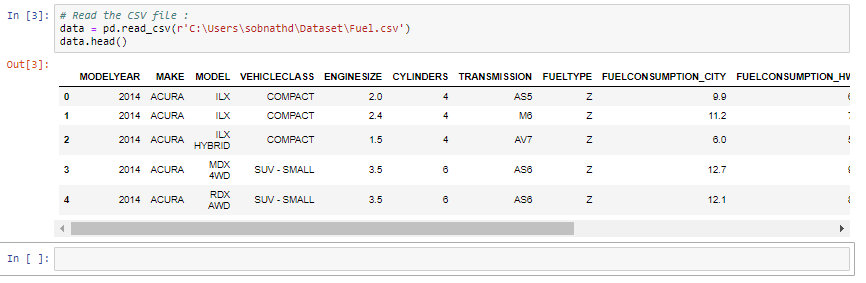
1. Next you need to read the CSV file that we will be using.

Download the **Fuel.csv** file from SOL and save it on your computer. Please make sure you know the path to the file. We check the first five rows of our dataset. In this case, we are using a vehicle model dataset.

Type the following code, add your own path to the file you downloaded and hit the **Run button**:

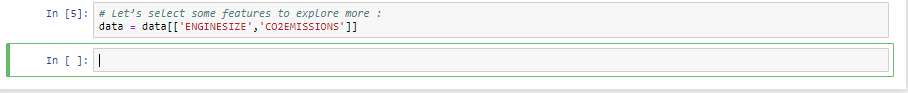


You should see the following screen:

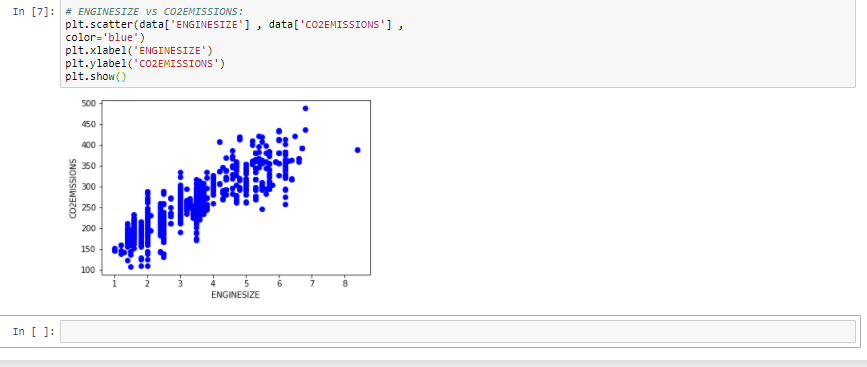


1. Now we need to select the features we want to consider in predicting values. Here our goal is to predict the value of “co2 emissions” from the value of “engine size” in our dataset.

Type the following codes and hit Run



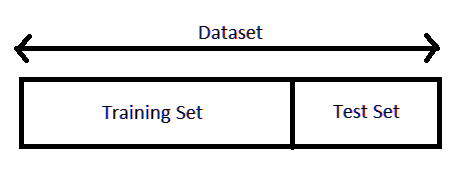
1. Now we want to plot this data and visualise this on a scatter plot. Type the following codes and hit Run.



Data plot for the linear regression algorithm

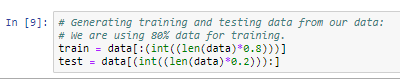
1. Divide the data into training and testing data

As we work with datasets, machine learning algorithms work in two stages. We usually split the data around 20%-80% between testing and training stages. Under supervised learning, we split a dataset into a training data and test data in Python ML.



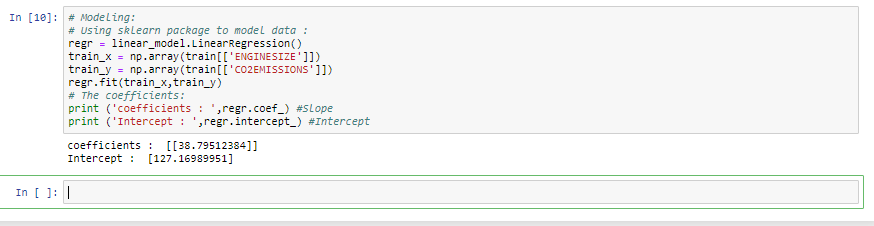
To check the accuracy of a model, we are going to divide our data into training and testing datasets. We will use training data to train our model, and then we will check the accuracy of our model using the testing dataset.

Type the following codes and hit Run.



1. Train the model

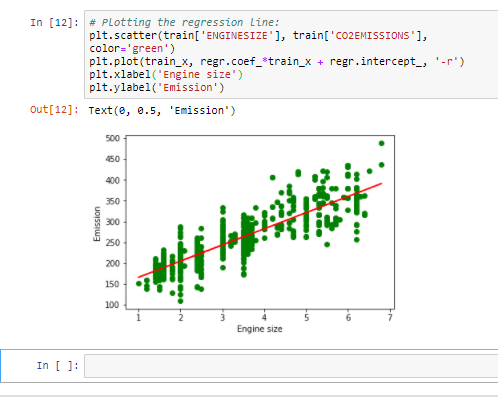
Here is how we can train our model and find the coefficients for our best-fit regression line. Once you type the following codes and hit Run, you should be able to see the coefficients and intercept.



This will make more sense when we plot this in a graph. We can replace the values in our linear equation.

Y= 38.795X + 127.17

1. Plot the best fit line. Type the following codes and hit Run.

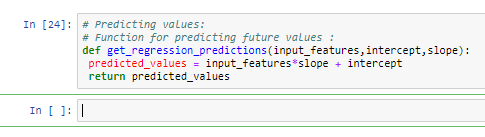


Data plot for linear regression based on its coefficients

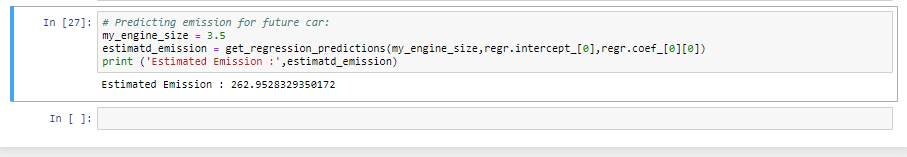
1. Prediction function

We are going to use a prediction function for our testing dataset. Predicting values of CO2 will be based on the regression line.

Type the following codes and hit Run:



Now let us predict the emission for an engine size of 3.5. Type the following codes and hit Run.



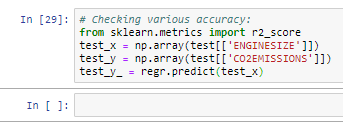
The CO2 prediction for a 3.5 engine size is 262.95 according to our model and algorithms.

1. Checking accuracy for test data

Now that we have the function for predicting the CO2 emission based on Engine size, we need to check for the accuracy of our model. We can check the accuracy of a model by comparing the actual values with the

predicted values in our dataset.

Type the following codes and hit Run.



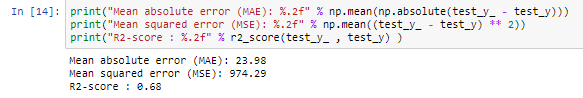
Let’s now print the results using the following codes. You can copy and paste the codes and hit Run:

Python Codes:

print("Mean absolute error (MAE): %.2f" % np.mean(np.absolute(test\_y\_ - test\_y)))

print("Mean squared error (MSE): %.2f" % np.mean((test\_y\_ - test\_y) \*\* 2))

print("R2-score : %.2f" % r2\_score(test\_y\_ , test\_y) )



1. Interpretation of results

The coefficients describe the mathematical relationship between each independent variable and the dependent variable. The p-values for the coefficients indicate whether these relationships are statistically significant.

* Mean Absolute Error (MAE) is a model evaluation metric used with regression models. The mean absolute error of a model with respect to a test set is the mean of the absolute values of the individual prediction errors on over all instances in the test set. Each prediction error is the difference between the true value and the predicted value for the instance. It is the mean of the absolute value of the errors.
* A small MAE suggests the model is great at prediction, while a large MAE suggests that your model may have trouble in certain areas. A MAE of 0 means that your model is a perfect predictor of the outputs (but this will almost never happen).
* Mean Squared Error (MSE): Mean Squared Error (MSE) is the mean of the squared error. It’s more popular than Mean Absolute Error because the focus is geared more towards large errors. This is due to the squared term exponentially increasing larger errors in comparison to smaller ones. MSE is always a positive value.
* R-squared (R2) is a goodness-of-fit measure for linear regression models. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively. The higher the R-squared, the better the model fits your data. Best possible score is 1.0. Here it is 0.68 which represent 68%.

Summary:

Mean Absolute Error (MAE): This measures the absolute average distance between the real data and the predicted data, but it fails to punish large errors in prediction.

Mean Square Error (MSE): This measures the squared average distance between the real data and the predicted data. A larger MSE means that the data values are dispersed widely around its central moment (mean), and a smaller MSE shows that your data values are dispersed closely to its central moment (mean).

R-squared is a goodness-of-fit measure for linear regression models. For example, an r-squared of 0.60 reveals that 60% of the data fit the regression model. Generally, a higher r-squared indicates a better fit for the model.