Teammates:

Sukumar Bodapati – sb5zh@umsystem.edu - 16326105

GitHub link: https://github.com/UMKC-APL-BigDataAnalytics/icp-8--assignment-8-sukumarbodapati

Sri Nikhitha Boddapati – sb4dz@umsystem.edu - 16322565

GitHub link: https://github.com/UMKC-APL-BigDataAnalytics/icp-8--assignment-8-Srinikhitha98

Video link: https://youtu.be/w1T9Do7h8E0

ICP - Big Data App & Analytics – Simple Linear Regression

Learnings in this ICP:

In this ICP, we have learned about the simple linear Regression model.

- A linear regression model predicts the relationship between the independent and dependent variable. Mathematically, y = *mx+ b* (y is the dependent variable, and x is the independent variable, m=slope). But generally, the data will not be linear, we will have cluster data. So, we predict the best fit line for the dataset.
- **Optimized Slope and Intercept:** The slope is nothing but the rate of change. When all the predictor variables in a regression model are equal to zero, the intercept (also known as the "constant") indicates the mean value of the response variable. We must calculate the optimized slope and intercept values to reduce the error.

Evaluation Metrics:

1. ***Mean absolute error***: This gives the mean of the absolute value of the errors. We can get the average error for our dataset
2. ***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.
3. **R-2Score:** R-squared is not error but is a popular metric for accuracy of your model. It represents how close the data are to the fitted regression line. The higher the R-squared, the better the model fits your data. Best possible score is 1.0 and it can be negative (because the model can be arbitrarily worse).

Objective:

We need to build a linear regression model for any data set and calculate the intercept, slope and mean square error.

Challenges:

1. The important issue in this model is we are assuming the linearity between the independent and dependent variables.
2. **Outliners:** In this model, the main problem is outliners since it is the linear regression. So, outliers should be analyzed and removed before applying Linear Regression to the dataset.

Tasks:

1. Import all the libraries required for the model. Import the linear model from the sklearn.

2. In this ICP, we are using the fuel consumption dataset which has fuel consumption and estimated CO2emission.

```
[ ] #Reading the dataset using the pandas
    df = pd.read_csv("FuelConsumption.csv")

    #Look at the dataset
    df.head()
```

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HWY	FUELCONSUMPTION_COMB	FUELCONSUMPTION_COMB_MPG	CO2EMISSIONS
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	9.9	6.7	8.5	33	196
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	11.2	7.7	9.6	29	221
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	6.0	5.8	5.9	48	136
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	12.7	9.1	11.1	25	255
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	12.1	8.7	10.6	27	244

3. In order to work with this data,lets know the statistics of the data.

```
[ ] # Summarize the data
    df.describe()
```

	MODELYEAR	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_CITY	FUELCONSUMPTION_HWY	FUELCONSUMPTION_COMB	FUELCONSUMPTION_COMB_MPG	CO2EMISSIONS
count	1067.0	1067.000000	1067.000000	1067.000000	1067.000000	1067.000000	1067.000000	1067.000000
mean	2014.0	3.346298	5.794752	13.296532	9.474602	11.580881	26.441425	256.228679
std	0.0	1.415895	1.797447	4.101253	2.794510	3.485595	7.468702	63.372304
min	2014.0	1.000000	3.000000	4.600000	4.900000	4.700000	11.000000	108.000000
25%	2014.0	2.000000	4.000000	10.250000	7.500000	9.000000	21.000000	207.000000
50%	2014.0	3.400000	6.000000	12.600000	8.800000	10.900000	26.000000	251.000000
75%	2014.0	4.300000	8.000000	15.550000	10.850000	13.350000	31.000000	294.000000
max	2014.0	8.400000	12.000000	30.200000	20.500000	25.800000	60.000000	488.000000

Data Exploration:

1. We have taken some of the features from the entire dataset. We have considered Engine size, cylinders,Fuelconsumption_combo,CO2 Emission as the features.

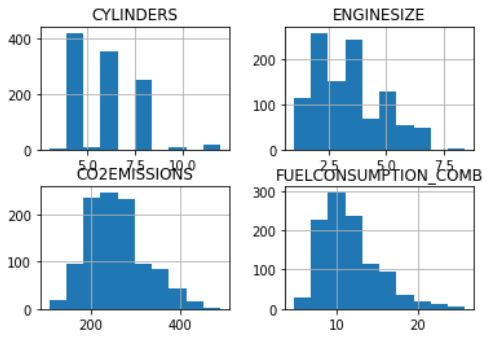
```
[ ] cdf = df[['ENGINESIZE','CYLINDERS','FUELCONSUMPTION_COMB','CO2EMISSIONS']]
    #Print the first 9 rows
    cdf.head(9)
```

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

Visualization of the dataset:

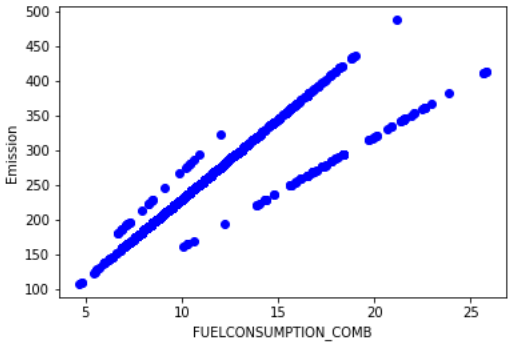
We have plotted the bargraphs for the features.

```
[ ] vis = cdf[['CYLINDERS','ENGINESIZE','CO2EMISSIONS','FUELCONSUMPTION_COMB']]
    vis.hist()
    plt.show()
```



1. **FUELCONSUMPTION_COMB** vs the Emission: Plot fuel consumption vs Emission to find the how linear they are:

```
[ ]  #Let's plot the Fuel consumption vs Emission
     plt.scatter(cdf.FUELCONSUMPTION_COMB, cdf.CO2EMISSIONS,  color='blue')
     plt.xlabel("FUELCONSUMPTION_COMB")
     plt.ylabel("Emission")
     plt.show()
```

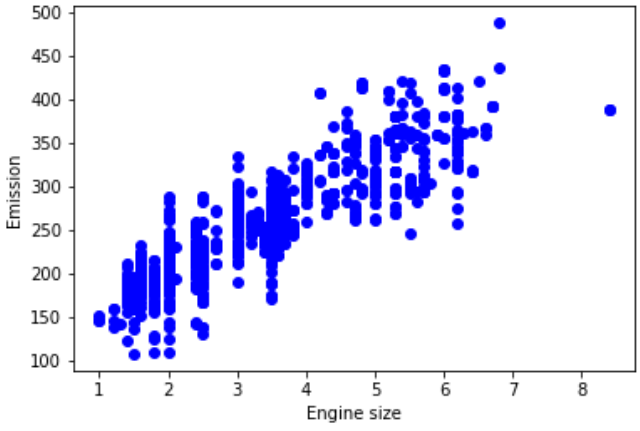


If we see the above graph ,we can their relation is almost linear.

2. **Engine size** vs the Emission:

Plot Engine size vs Emission to find the how linear they are:

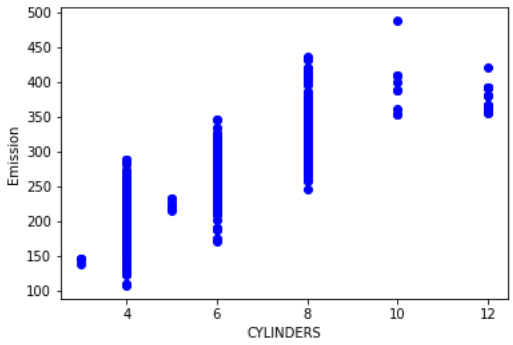
```
[ ]  #Let's plot the Engine size vs Emission
     plt.scatter(cdf.ENGINESIZE, cdf.CO2EMISSIONS,  color='blue')
     plt.xlabel("Engine size")
     plt.ylabel("Emission")
     plt.show()
```



If we observe the above graph, we can see the data is clustered.

3. **CYLINDER** vs the Emission:
   Plot **CYLINDER** vs Emission to find the how linear they are:

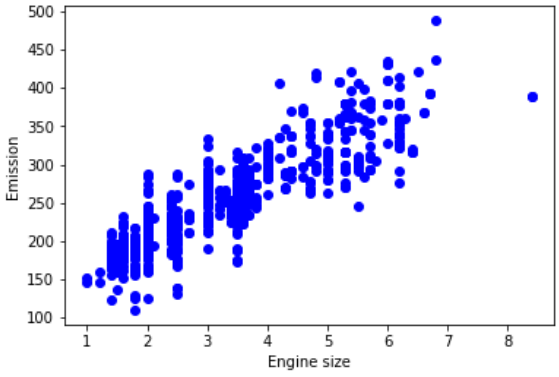
```
[ ] plt.scatter(cdf.CYLINDERS, cdf.CO2EMISSIONS,  color='blue')
    plt.xlabel("CYLINDERS")
    plt.ylabel("Emission")
    plt.show()
```



We have taken 80% of the data for the training and 20 % of the data for testing.From the above plot,if we observe the Engine size vs Emissions graph, we have clustered data.We have considered Engine size and Emission as dependent & Independent variables.

Train data graph:

```
[ ] plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
    plt.xlabel("Engine size")
    plt.ylabel("Emission")
    plt.show()
```



Modeling the Dataset:

Building the linear regression model and Calculating the slope and intercept of the data.We can directly calculate using the sklearn.

```
[51] regr = linear_model.LinearRegression()
     train_x = np.asanyarray(train[['ENGINESIZE']])
     train_y = np.asanyarray(train[['CO2EMISSIONS']])
     regr.fit (train_x, train_y)
     # The coefficients
     print ('Slope of the model: ', regr.coef_)
     print ('Intercept of the model: ',regr.intercept_)

     Slope of the model:  [[39.36264979]]
     Intercept of the model:  [124.77824609]
```

Assign the x_train ,y_train with the respective variable (x=Engine size and y=CO2Emissions).Fit the model using the trained datasets.Calculate the slope using the coef_ and intercept using the intercept_.

The value for this model are:

Slope of the Model: 39.3

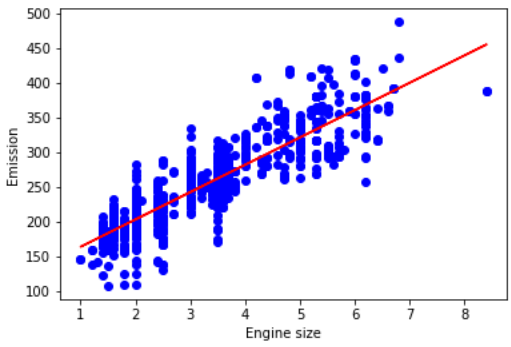
Intercept of the Model :124.58

Plot the output:

Regression line = (regr.coef_)* training data + regr.intercept.(y=mx+c)

```
[52] plt.scatter(train.ENGINESIZE, train.CO2EMISSIONS,  color='blue')
     plt.plot(train_x, regr.coef_[0][0]*train_x + regr.intercept_[0], '-r')
     plt.xlabel("Engine size")
     plt.ylabel("Emission")

     Text(0, 0.5, 'Emission')
```



Plot the regression line and training data.

To decide whether this is the best fit line or not, we have to calculate the error.

Evaluation of the model:

We have to compare the actual value and predicted values to calculate the accuracy of the model.In order to evaluate,we have to calculate the following errors:

Mean Absolute error: This is the mean of the absolute values of the errors. The mean absolute error is **24.17.**

Mean Squared Error (MSE): Mean Squared Error (MSE) is the mean of the squared error. The MSE of the model is **901.11.**

R2-Score: Calculate the R2-Score and the R2-Score is 0.70 for this model

```
[53] test_x = np.asanyarray(test[['ENGINESIZE']])
     test_y = np.asanyarray(test[['CO2EMISSIONS']])
     test_y_hat = regr.predict(test_x)

     print("Mean absolute error: %.2f" % np.mean(np.absolute(test_y_hat - test_y)))
     print("Residual sum of squares (MSE): %.2f" % np.mean((test_y_hat - test_y) ** 2))
     print("R2-score: %.2f" % r2_score(test_y_hat , test_y) )
```

```
Mean absolute error: 22.56
Residual sum of squares (MSE): 901.11
R2-score: 0.70
```

The R2-score is 0.7 which says that the model predicts the best fit line.