Multiple Linear Regression

In [14]:
```
import pandas as pd
from sklearn import linear_model
import statsmodels.api as sm
from scipy import stats
```

In [26]:
```
df=pd.read_csv("C:/Users/SAGNIK SAMANTA/Downloads/data.csv")
df.head()
```

Out[26]:

	Car	Model	Volume	Weight	CO2
0	Toyoty	Aygo	1000	790	99
1	Mitsubishi	Space Star	1200	1160	95
2	Skoda	Citigo	1000	929	95
3	Fiat	500	900	865	90
4	Mini	Cooper	1500	1140	105

We can predict the CO2 emission of a car based on the size of the engine, but with multiple regression we can throw in more variables, like the weight of the car, to make the prediction more accurate.

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   Car     36 non-null     object
 1   Model   36 non-null     object
 2   Volume  36 non-null     int64 
 3   Weight  36 non-null     int64 
 4   CO2     36 non-null     int64 
dtypes: int64(3), object(2)
memory usage: 1.5+ KB
```

In [17]:
```
df.columns
```

Out[17]:
```
Index(['Car', 'Model', 'Volume', 'Weight', 'CO2'], dtype='object')
```

In [22]:
```
X=df[['Weight', 'Volume']]
y=df["CO2"]
```

From the sklearn module we will use the LinearRegression() method to create a linear regression object.

This object has a method called fit() that takes the independent and dependent values as parameters and fills the regression object with data that describes the

relationship:

In [23]:
```
reg_model=linear_model.LinearRegression()
reg_model.fit(X,y)
```

Out[23]:
```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

Now we have a regression object that are ready to predict CO2 values based on a car's weight and volume:

predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3 :

In [7]:
```
Predicted_CO2=reg_model.predict([[2300,1300]])
print("Predicted_CO2 : ",Predicted_CO2)
```

```
Predicted_CO2 :  [107.2087328]

C:\Users\SAGNIK SAMANTA\anaconda3\Lib\site-packages\sklearn\base.py:464:
UserWarning: X does not have valid feature names, but LinearRegression wa
s fitted with feature names
  warnings.warn(
```

Print the coefficient values of the regression object :

In [8]:
```
print(reg_model.coef_)
```

```
[0.00755095 0.00780526]
```

The result array represents the coefficient values of weight and volume.

Weight: 0.00755095 Volume: 0.00780526

These values tell us that if the weight increase by 1kg, the CO2 emission increases by 0.00755095g.

And if the engine size (Volume) increases by 1 cm3, the CO2 emission increases by 0.00780526 g.

I think that is a fair guess, but let test it!

We have already predicted that if a car with a 1300cm3 engine weighs 2300kg, the CO2 emission will be approximately 107g.

What if we increase the weight with 1000kg?

In [9]:
```
Predicted_CO2=reg_model.predict([[3300,1300]])
print("Predicted_CO2 :",Predicted_CO2)
```

```
Predicted_CO2 : [114.75968007]

C:\Users\SAGNIK SAMANTA\anaconda3\Lib\site-packages\sklearn\base.py:464:
UserWarning: X does not have valid feature names, but LinearRegression wa
s fitted with feature names
  warnings.warn(
```

We have predicted that a car with 1.3 liter engine, and a weight of 3300 kg, will release approximately 115 grams of CO2 for every kilometer it drives.

Which shows that the coefficient of 0.00755095 is correct:

107.2087328 + (1000 * 0.00755095) = 114.75968

In [24]:
```
X2 = sm.add_constant(X)
Reg_model = sm.OLS(y, X2)
Reg = Reg_model.fit()
print(Reg.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    CO2   R-squared:                       0.377
Model:                            OLS   Adj. R-squared:                  0.339
Method:                 Least Squares   F-statistic:                     9.966
Date:                Mon, 01 Apr 2024   Prob (F-statistic):           0.000411
Time:                        13:56:10   Log-Likelihood:                -114.39
No. Observations:                  36   AIC:                             234.8
Df Residuals:                      33   BIC:                             239.5
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         79.6947      5.564     14.322      0.000      68.374      91.016
Weight         0.0076      0.006      1.173      0.249      -0.006       0.021
Volume         0.0078      0.004      1.948      0.060      -0.000       0.016
==============================================================================
Omnibus:                        4.957   Durbin-Watson:                   0.944
Prob(Omnibus):                  0.084   Jarque-Bera (JB):                1.836
Skew:                          -0.025   Prob(JB):                        0.399
Kurtosis:                       1.895   Cond. No.                     1.16e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.16e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [ ]: