# Assignment No 2

## Problem Statement -

Perform the following operations using Python on the Air quality data sets

a. Data cleaning

b. Data integration

c. Data transformation

d. Error correcting

e. Data model building

# Importing required libraries

In [58]:

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
```

# Reading dataset

In [59]:
```
data = pd.read_csv("airquality2.csv")
data
```

Out[59]:

| | Unnamed: 0 | Ozone | Solar.R | Wind | Temp | Month | Day | Humidity |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 41.0 | 190.0 | 7.4 | 67 | 5 | 1 | High |
| **1** | 2 | 36.0 | 118.0 | 8.0 | 72 | 5 | 2 | High |
| **2** | 3 | 12.0 | 149.0 | 12.6 | 74 | 5 | 3 | NaN |
| **3** | 4 | 18.0 | 313.0 | 11.5 | 62 | 5 | 4 | Medium |
| **4** | 5 | NaN | NaN | 14.3 | 56 | 5 | 5 | Low |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... |
| **148** | 149 | 30.0 | 193.0 | 6.9 | 70 | 9 | 26 | Low |
| **149** | 150 | NaN | 145.0 | 13.2 | 77 | 9 | 27 | Low |
| **150** | 151 | 14.0 | 191.0 | 14.3 | 75 | 9 | 28 | High |
| **151** | 152 | 18.0 | 131.0 | 8.0 | 76 | 9 | 29 | High |
| **152** | 153 | 20.0 | 223.0 | 11.5 | 68 | 9 | 30 | Medium |

153 rows × 8 columns

In [60]:
```
data.drop(data.iloc[:,[0]], axis=1, inplace=True)
data
```

Out[60]:

| | Ozone | Solar.R | Wind | Temp | Month | Day | Humidity |
|---|---|---|---|---|---|---|---|
| **0** | 41.0 | 190.0 | 7.4 | 67 | 5 | 1 | High |
| **1** | 36.0 | 118.0 | 8.0 | 72 | 5 | 2 | High |
| **2** | 12.0 | 149.0 | 12.6 | 74 | 5 | 3 | NaN |
| **3** | 18.0 | 313.0 | 11.5 | 62 | 5 | 4 | Medium |
| **4** | NaN | NaN | 14.3 | 56 | 5 | 5 | Low |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **148** | 30.0 | 193.0 | 6.9 | 70 | 9 | 26 | Low |
| **149** | NaN | 145.0 | 13.2 | 77 | 9 | 27 | Low |
| **150** | 14.0 | 191.0 | 14.3 | 75 | 9 | 28 | High |
| **151** | 18.0 | 131.0 | 8.0 | 76 | 9 | 29 | High |
| **152** | 20.0 | 223.0 | 11.5 | 68 | 9 | 30 | Medium |

153 rows × 7 columns

In [61]:
```
data.isnull().sum()
```

Out[61]:
```
Ozone       37
Solar.R      7
Wind         0
Temp         0
Month        0
Day          0
Humidity     9
dtype: int64
```

In [62]:
```
data["Ozone"].fillna(data["Ozone"].mean(), inplace=True)
data["Solar.R"].fillna(data["Solar.R"].mean(), inplace=True)
data
```

Out[62]:

| | Ozone | Solar.R | Wind | Temp | Month | Day | Humidity |
|---|---|---|---|---|---|---|---|
| **0** | 41.00000 | 190.000000 | 7.4 | 67 | 5 | 1 | High |
| **1** | 36.00000 | 118.000000 | 8.0 | 72 | 5 | 2 | High |
| **2** | 12.00000 | 149.000000 | 12.6 | 74 | 5 | 3 | NaN |
| **3** | 18.00000 | 313.000000 | 11.5 | 62 | 5 | 4 | Medium |
| **4** | 42.12931 | 185.931507 | 14.3 | 56 | 5 | 5 | Low |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **148** | 30.00000 | 193.000000 | 6.9 | 70 | 9 | 26 | Low |
| **149** | 42.12931 | 145.000000 | 13.2 | 77 | 9 | 27 | Low |
| **150** | 14.00000 | 191.000000 | 14.3 | 75 | 9 | 28 | High |
| **151** | 18.00000 | 131.000000 | 8.0 | 76 | 9 | 29 | High |
| **152** | 20.00000 | 223.000000 | 11.5 | 68 | 9 | 30 | Medium |

153 rows × 7 columns

In [63]:
```
data["Humidity"].fillna("Medium", inplace=True)
```

In [64]:
```
data.isnull().sum()
```

Out[64]:
```
Ozone       0
Solar.R     0
Wind        0
Temp        0
Month       0
Day         0
Humidity    0
dtype: int64
```

# Performing Label Encoding

In [65]:
```
from sklearn.preprocessing import LabelEncoder
```

In [66]:
```
le = LabelEncoder()
```

In [67]:
```
data["Humidity"] = le.fit_transform(data["Humidity"])
```

In [68]:
```
data
```

Out[68]:

| | Ozone | Solar.R | Wind | Temp | Month | Day | Humidity |
|---|---|---|---|---|---|---|---|
| **0** | 41.00000 | 190.000000 | 7.4 | 67 | 5 | 1 | 0 |
| **1** | 36.00000 | 118.000000 | 8.0 | 72 | 5 | 2 | 0 |
| **2** | 12.00000 | 149.000000 | 12.6 | 74 | 5 | 3 | 2 |
| **3** | 18.00000 | 313.000000 | 11.5 | 62 | 5 | 4 | 2 |
| **4** | 42.12931 | 185.931507 | 14.3 | 56 | 5 | 5 | 1 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **148** | 30.00000 | 193.000000 | 6.9 | 70 | 9 | 26 | 1 |
| **149** | 42.12931 | 145.000000 | 13.2 | 77 | 9 | 27 | 1 |
| **150** | 14.00000 | 191.000000 | 14.3 | 75 | 9 | 28 | 0 |
| **151** | 18.00000 | 131.000000 | 8.0 | 76 | 9 | 29 | 0 |
| **152** | 20.00000 | 223.000000 | 11.5 | 68 | 9 | 30 | 2 |

153 rows × 7 columns

# Assigning dependent and independent variables

In [69]:
```
x = data.iloc[:, [0,1,2,3]]
y = data["Humidity"]
```

# Importing the required function and splitting the data into training and testing data

In [70]:
```
from sklearn.model_selection import train_test_split
```

In [71]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

# Printing size of training and testing data

In [72]:
```
print(len(x_train))
print(len(x_test))
print(len(y_train))
print(len(y_test))
```

```
122
31
122
31
```

# Importing and creating a linear regression model

In [73]:
```
from sklearn import linear_model
```

In [74]:
```
regr = linear_model.LinearRegression()
```

# Fitting the model and display the regression coefficients

In [75]:
```
model = regr.fit(x_train, y_train)
print(model.intercept_)
print(model.coef_)
```

```
0.2596553365337608
[-0.00196037 -0.00092793  0.02911026  0.01001252]
```

# Predicting the values

In [76]:
```
y_predict = model.predict(x_test)
```

In [77]:
```
y_predict
```

Out[77]:
```
array([1.07433468, 1.09123502, 0.79142388, 1.13114234, 0.94300376,
       1.11784014, 1.09664385, 0.80813102, 0.89763294, 1.12174905,
       0.9709549 , 1.00403062, 1.07364943, 1.2003274 , 1.10474293,
       1.16587009, 1.07330507, 1.17589964, 1.24793266, 0.99615301,
       0.9414399 , 1.18835405, 1.04807664, 1.19328169, 0.97611117,
       0.70579647, 1.01100268, 0.90005967, 1.19916514, 1.03337014,
       1.06991779])
```

# Importing and calculating the performance metrics

In [78]:
```
from sklearn.metrics import mean_squared_error
```

## Mean Squared Error (MSE)

In [79]:
```
mse = mean_squared_error(y_test,y_predict)
mse
```

Out[79]:
```
0.7555852807566938
```

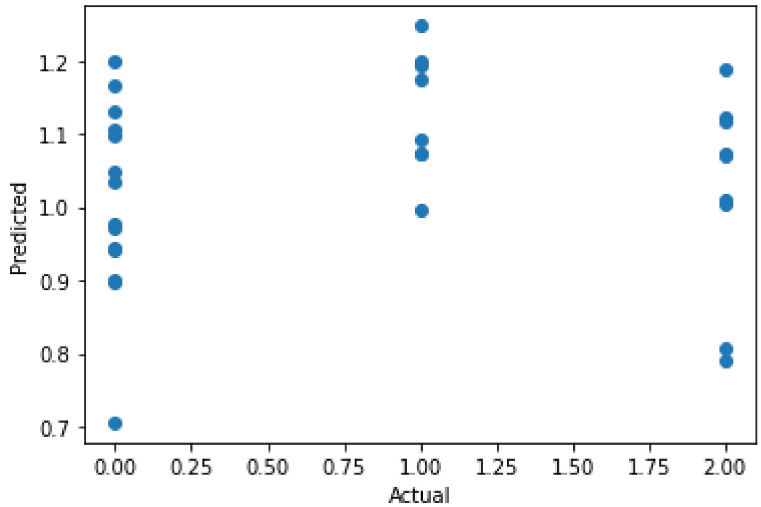
## Root Mean Squared Error (RMSE)

In [80]:
```
rmse = np.sqrt(mse)
rmse
```

Out[80]:
```
0.8692440858336017
```

# Plotting the data

In [83]:
```
plt.scatter(y_test,y_predict);
plt.xlabel('Actual');
plt.ylabel('Predicted');
```




In [85]:
```
sns.regplot(x=y_test,y=y_predict,ci=None,color ='red');
```