From this websiet this dataset is collected

https://cpcb.nic.in/

Import Necessary Libraries

In [359…

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

Load the dataset

In [360…

```
df = pd.read_csv('city_day.csv')
df.head()
```

Out[360]:

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	133.36	0.00	0.02	0.00	NaN	NaN
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	34.06	3.68	5.50	3.77	NaN	NaN
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	30.70	6.80	16.40	2.25	NaN	NaN
3	Ahmedabad	2015-01-04	NaN	NaN	1.70	18.48	17.97	NaN	1.70	18.59	36.08	4.43	10.14	1.00	NaN	NaN
4	Ahmedabad	2015-01-05	NaN	NaN	22.10	21.42	37.76	NaN	22.10	39.33	39.31	7.01	18.89	2.78	NaN	NaN

In [361…

```
df.shape
```

Out[361]:

```
(29531, 16)
```

In [362…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29531 entries, 0 to 29530
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   City        29531 non-null  object 
 1   Date        29531 non-null  object 
 2   PM2.5       24933 non-null  float64
 3   PM10        18391 non-null  float64
 4   NO          25949 non-null  float64
 5   NO2         25946 non-null  float64
 6   NOx         25346 non-null  float64
 7   NH3         19203 non-null  float64
 8   CO          27472 non-null  float64
 9   SO2         25677 non-null  float64
 10  O3          25509 non-null  float64
 11  Benzene     23908 non-null  float64
 12  Toluene     21490 non-null  float64
 13  Xylene      11422 non-null  float64
 14  AQI         24850 non-null  float64
 15  AQI_Bucket  24850 non-null  object 
dtypes: float64(13), object(3)
memory usage: 3.6+ MB
```

Removing Null value rows

In [363…

```
df = df.dropna(axis=0)
```

In [364…

```
df
```

Out[364]:

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
2123	Amaravati	2017-11-25	81.40	124.50	1.44	20.50	12.08	10.72	0.12	15.24	127.09	0.20	6.50	0.06	184.0	Moderate
2124	Amaravati	2017-11-26	78.32	129.06	1.26	26.00	14.85	10.28	0.14	26.96	117.44	0.22	7.95	0.08	197.0	Moderate
2125	Amaravati	2017-11-27	88.76	135.32	6.60	30.85	21.77	12.91	0.11	33.59	111.81	0.29	7.63	0.12	198.0	Moderate
2126	Amaravati	2017-11-28	64.18	104.09	2.56	28.07	17.01	11.42	0.09	19.00	138.18	0.17	5.02	0.07	188.0	Moderate
2127	Amaravati	2017-11-29	72.47	114.84	5.23	23.20	16.59	12.25	0.16	10.55	109.74	0.21	4.71	0.08	173.0	Moderate
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29525	Visakhapatnam	2020-06-26	7.63	32.27	5.91	23.27	17.19	11.15	0.46	6.87	19.90	1.45	5.37	1.45	47.0	Good
29526	Visakhapatnam	2020-06-27	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.30	2.24	12.07	0.73	41.0	Good
29527	Visakhapatnam	2020-06-28	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.14	0.74	2.21	0.38	70.0	Satisfactory
29528	Visakhapatnam	2020-06-29	22.91	65.73	3.45	29.53	18.33	10.71	0.48	8.42	30.96	0.01	0.01	0.00	68.0	Satisfactory
29529	Visakhapatnam	2020-06-30	16.64	49.97	4.05	29.26	18.80	10.03	0.52	9.84	28.30	0.00	0.00	0.00	54.0	Satisfactory

6236 rows × 16 columns

Checking! Now all null value rows are removed

In [365…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6236 entries, 2123 to 29529
Data columns (total 16 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   City        6236 non-null   object 
 1   Date        6236 non-null   object 
 2   PM2.5       6236 non-null   float64
 3   PM10        6236 non-null   float64
 4   NO          6236 non-null   float64
 5   NO2         6236 non-null   float64
 6   NOx         6236 non-null   float64
 7   NH3         6236 non-null   float64
 8   CO          6236 non-null   float64
 9   SO2         6236 non-null   float64
 10  O3          6236 non-null   float64
 11  Benzene     6236 non-null   float64
 12  Toluene     6236 non-null   float64
 13  Xylene      6236 non-null   float64
 14  AQI         6236 non-null   float64
 15  AQI_Bucket  6236 non-null   object 
dtypes: float64(13), object(3)
memory usage: 828.2+ KB
```

Removing date column because it has all unique values which is not useful for decision tree

In [366…

```
df.nunique()
```

Out[366]:

```
City             9
Date          1828
PM2.5         4769
PM10          5399
NO            2966
NO2           3976
NOx           3962
NH3           3248
CO             423
SO2           2169
O3            3957
Benzene       1203
Toluene       2446
Xylene        1036
AQI            452
AQI_Bucket       6
dtype: int64
```

In [367…

```
df = df.drop("Date", axis=1)
```

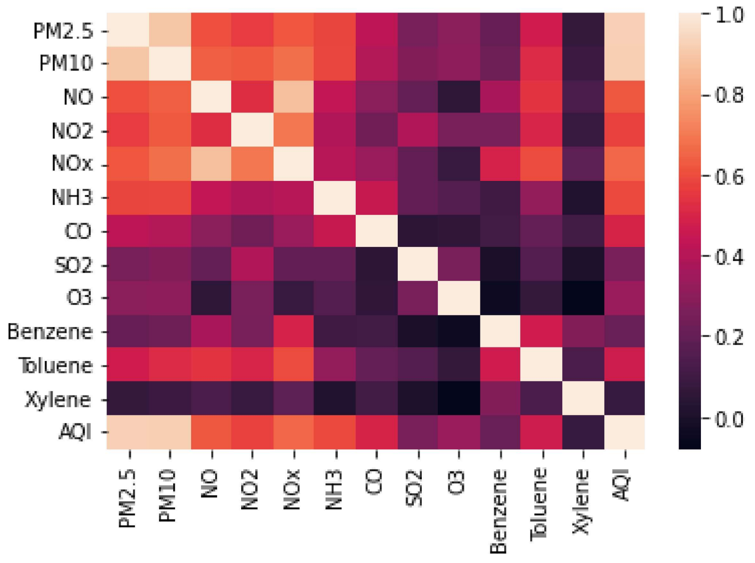
Visualizing Correlation Matrix

In [368…

```
corr = df.corr()
```

In [369…

```
sns.heatmap(corr);
```



- #### from above chart it is clear that AQI is highly dependent on PM2.5 and PM10

Final Dataframe

In [371…

```
df
```

Out[371]:

	City	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	O3	Benzene	Toluene	Xylene	AQI	AQI_Bucket
2123	Amaravati	81.40	124.50	1.44	20.50	12.08	10.72	0.12	15.24	127.09	0.20	6.50	0.06	184.0	Moderate
2124	Amaravati	78.32	129.06	1.26	26.00	14.85	10.28	0.14	26.96	117.44	0.22	7.95	0.08	197.0	Moderate
2125	Amaravati	88.76	135.32	6.60	30.85	21.77	12.91	0.11	33.59	111.81	0.29	7.63	0.12	198.0	Moderate
2126	Amaravati	64.18	104.09	2.56	28.07	17.01	11.42	0.09	19.00	138.18	0.17	5.02	0.07	188.0	Moderate
2127	Amaravati	72.47	114.84	5.23	23.20	16.59	12.25	0.16	10.55	109.74	0.21	4.71	0.08	173.0	Moderate
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
29525	Visakhapatnam	7.63	32.27	5.91	23.27	17.19	11.15	0.46	6.87	19.90	1.45	5.37	1.45	47.0	Good
29526	Visakhapatnam	15.02	50.94	7.68	25.06	19.54	12.47	0.47	8.55	23.30	2.24	12.07	0.73	41.0	Good
29527	Visakhapatnam	24.38	74.09	3.42	26.06	16.53	11.99	0.52	12.72	30.14	0.74	2.21	0.38	70.0	Satisfactory
29528	Visakhapatnam	22.91	65.73	3.45	29.53	18.33	10.71	0.48	8.42	30.96	0.01	0.01	0.00	68.0	Satisfactory
29529	Visakhapatnam	16.64	49.97	4.05	29.26	18.80	10.03	0.52	9.84	28.30	0.00	0.00	0.00	54.0	Satisfactory

6236 rows × 15 columns

Seperating Input and output columns

In [372...

```
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

Converting Categorical data into numerical data

In [373...

```
X = pd.get_dummies(data=X, columns=['City'], drop_first=True)
```

In [374...

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

Train test split

In [375...

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3)
```

Decision Tree

In [376...

```
from sklearn.tree import DecisionTreeClassifier
tree = DecisionTreeClassifier()
tree.fit(X_train,y_train)
```

Out[376]:

```
DecisionTreeClassifier()
```

In [377...

```
pred = tree.predict(X_test)
```

In [378...

```
accuracy_score(y_test, pred)
```

Out[378]:

```
1.0
```

Confusion Matrix

In [379...

```
from sklearn.metrics import confusion_matrix,classification_report
confusion_matrix(y_test, pred)
```

Out[379]:

```
array([[145,   0,   0,   0,   0,   0],
       [  0, 742,   0,   0,   0,   0],
       [  0,   0, 189,   0,   0,   0],
       [  0,   0,   0, 635,   0,   0],
       [  0,   0,   0,   0,  37,   0],
       [  0,   0,   0,   0,   0, 123]], dtype=int64)
```

Classification Report

In [380...

```
print(classification_report(y_test, pred, digits=3))
```

```
              precision    recall  f1-score   support

           0      1.000     1.000     1.000       145
           1      1.000     1.000     1.000       742
           2      1.000     1.000     1.000       189
           3      1.000     1.000     1.000       635
           4      1.000     1.000     1.000        37
           5      1.000     1.000     1.000       123

    accuracy                          1.000      1871
   macro avg      1.000     1.000     1.000      1871
weighted avg      1.000     1.000     1.000      1871
```

Saving the model

In [381…

```python
import pickle
```

In [382…

```python
with open('model_pkl', 'wb') as files:
    pickle.dump(tree, files)
```

Test Your Inputs

In [383…

```python
with open('model_pkl' , 'rb') as f:
    tree = pickle.load(f)
```

In [385…

```python
inps = []
inps.append(float(input("PM2.5:- ")))
inps.append(float(input('PM10:- ')))
inps.append(float(input('NO:- ')))
inps.append(float(input('NO2:- ')))
inps.append(float(input( 'NOx:- ')))
inps.append(float(input('NH3:- ')))
inps.append(float(input('CO:- ')))
inps.append(float(input('SO2:- ')))
inps.append(float(input('O3:- ')))
inps.append(float(input('Benzene:- ')))
inps.append(float(input('Toluene:- ')))
inps.append(float(input('Xylene:- ')))
inps.append(float(input('AQI:- ')))

city = int(input("Select You City 0=Amaravati, 1=Amritsar, 2=Chandigarh, 3=Delhi, 4=Gurugram,5=Hyderabad, 6=Kolkata, 7=Patna, 8s=V
if city==0:
    inps+=[0.,0.,0.,0.,0.,0.,0.,0.]
elif city==1:
    inps+=[1.,0.,0.,0.,0.,0.,0.,0.]
elif city==2:
    inps+=[0.,1.,0.,0.,0.,0.,0.,0.]
elif city==3:
    inps+=[0.,0.,1.,0.,0.,0.,0.,0.]
elif city==4:
    inps+=[0.,0.,0.,1.,0.,0.,0.,0.]
elif city==5:
    inps+=[0.,0.,0.,0.,1.,0.,0.,0.]
elif city==6:
    inps+=[0.,0.,0.,0.,0.,1.,0.,0.]
elif city==7:
    inps+=[0.,0.,0.,0.,0.,0.,1.,0.]
elif city==8:
    inps+=[0.,0.,0.,0.,0.,0.,0.,1.]
arr = ['Good', 'Moderate', 'Poor', 'Satisfactory', 'Severe', 'Very Poor']
arr[tree.predict([inps])[0]]
```

```
PM2.5:- 0
PM10:- 0
NO:- 0
NO2:- 0
NOx:- 0
NH3:- 0
CO:- 0
SO2:- 0
O3:- 0
Benzene:- 0
Toluene:- 0
Xylene:- 0
AQI:- 0
Select You City 0=Amaravati, 1=Amritsar, 2=Chandigarh, 3=Delhi, 4=Gurugram,5=Hyderabad, 6=Kolkata, 7=Patna, 8s=Visakhapatnam: 0
```

```
C:\Users\SBS05\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid
feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

Out[385]:

```
'Good'
```

Direct Predict

In [391…

```python
arr = ['Good', 'Moderate', 'Poor', 'Satisfactory', 'Severe', 'Very Poor']
arr[tree.predict([[ 26.12,  90.2 ,  13.03,  34.39,  28.89,   7.49,   0.88,  10.89,
         6.41,   4.4 ,   9.58,   3.09, 113.  ,   0.  ,   0.  ,   0.  ,
         0.  ,   0.  ,   0.  ,   0.  ,   1.  ]])[0]]
```

```
C:\Users\SBS05\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid
feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

Out[391]:

```
'Moderate'
```

In [389…

```
arr = ['Good', 'Moderate', 'Poor', 'Satisfactory', 'Severe', 'Very Poor']
arr[tree.predict([[float(i) for i in input().split(",")]])[0]]
```

```
26.12,  90.2 ,  13.03,  34.39,  28.89,   7.49,   0.88,  10.89,          6.41,   4.4 ,   9.58,   3.09, 113.  ,   0.  ,   0.  ,
0.  ,          0.  ,   0.  ,   0.  ,   0.  ,   1.
```

```
C:\Users\SBS05\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid
feature names, but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

Out[389]:

```
'Moderate'
```