In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
import re
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
```

In [2]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2003.csv")
a
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2003-03-01 01:00:00	NaN	1.72	NaN	NaN	NaN	73.900002	316.299988	NaN	10.550000	55.2
1	2003-03-01 01:00:00	NaN	1.45	NaN	NaN	0.26	72.110001	250.000000	0.73	6.720000	52.3
2	2003-03-01 01:00:00	NaN	1.57	NaN	NaN	NaN	80.559998	224.199997	NaN	21.049999	63.2
3	2003-03-01 01:00:00	NaN	2.45	NaN	NaN	NaN	78.370003	450.399994	NaN	4.220000	67.8
4	2003-03-01 01:00:00	NaN	3.26	NaN	NaN	NaN	96.250000	479.100006	NaN	8.460000	95.7
...	...	...	...	...	...	...	...	...	...	...	
243979	2003-10-01 00:00:00	0.20	0.16	2.01	3.17	0.02	31.799999	32.299999	1.68	34.049999	7.3
243980	2003-10-01 00:00:00	0.32	0.08	0.36	0.72	NaN	10.450000	14.760000	1.00	34.610001	7.4
243981	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	34.639999	50.810001	NaN	32.160000	16.8
243982	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	32.580002	41.020000	NaN	NaN	13.5
243983	2003-10-01 00:00:00	1.00	0.29	2.15	6.41	0.07	37.150002	56.849998	2.28	21.480000	12.3

243984 rows × 16 columns

In [3]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 243984 entries, 0 to 243983
Data columns (total 16 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     243984 non-null  object 
 1   BEN      69745 non-null   float64
 2   CO       225340 non-null  float64
 3   EBE      61244 non-null   float64
 4   MXY      42045 non-null   float64
 5   NMHC     111951 non-null  float64
 6   NO_2     242625 non-null  float64
 7   NOx      242629 non-null  float64
 8   OXY      42072 non-null   float64
 9   O_3      234131 non-null  float64
 10  PM10     240896 non-null  float64
 11  PXY      42063 non-null   float64
 12  SO_2     242729 non-null  float64
 13  TCH      111991 non-null  float64
 14  TOL      69439 non-null   float64
 15  station  243984 non-null  int64  
dtypes: float64(14), int64(1), object(1)
memory usage: 29.8+ MB
```

In [4]:

```
b=a.fillna(value=104)
b
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	
0	2003-03-01 01:00:00	104.00	1.72	104.00	104.00	104.00	73.900002	316.299988	104.00	10.5
1	2003-03-01 01:00:00	104.00	1.45	104.00	104.00	0.26	72.110001	250.000000	0.73	6.7
2	2003-03-01 01:00:00	104.00	1.57	104.00	104.00	104.00	80.559998	224.199997	104.00	21.0
3	2003-03-01 01:00:00	104.00	2.45	104.00	104.00	104.00	78.370003	450.399994	104.00	4.2
4	2003-03-01 01:00:00	104.00	3.26	104.00	104.00	104.00	96.250000	479.100006	104.00	8.4
...	...	...	...	...	...	...	...	...	...	
243979	2003-10-01 00:00:00	0.20	0.16	2.01	3.17	0.02	31.799999	32.299999	1.68	34.0
243980	2003-10-01 00:00:00	0.32	0.08	0.36	0.72	104.00	10.450000	14.760000	1.00	34.6
243981	2003-10-01 00:00:00	104.00	104.00	104.00	104.00	0.07	34.639999	50.810001	104.00	32.1
243982	2003-10-01 00:00:00	104.00	104.00	104.00	104.00	0.07	32.580002	41.020000	104.00	104.0
243983	2003-10-01 00:00:00	1.00	0.29	2.15	6.41	0.07	37.150002	56.849998	2.28	21.4

243984 rows × 16 columns

In [5]:

```
b.columns
```

Out[5]:

```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O
_3',
       'PM10', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [6]:

```
c=b.head(10)
c
```

Out[6]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_
0	2003-03-01 01:00:00	104.00	1.72	104.00	104.00	104.00	73.900002	316.299988	104.00	10.55000
1	2003-03-01 01:00:00	104.00	1.45	104.00	104.00	0.26	72.110001	250.000000	0.73	6.72000
2	2003-03-01 01:00:00	104.00	1.57	104.00	104.00	104.00	80.559998	224.199997	104.00	21.04999
3	2003-03-01 01:00:00	104.00	2.45	104.00	104.00	104.00	78.370003	450.399994	104.00	4.22000
4	2003-03-01 01:00:00	104.00	3.26	104.00	104.00	104.00	96.250000	479.100006	104.00	8.46000
5	2003-03-01 01:00:00	8.41	1.94	9.83	21.49	0.45	90.300003	384.899994	9.48	9.95000
6	2003-03-01 01:00:00	104.00	1.38	104.00	104.00	0.29	89.580002	230.000000	104.00	7.20000
7	2003-03-01 01:00:00	104.00	1.58	104.00	104.00	0.30	93.639999	334.600006	104.00	4.19000
8	2003-03-01 01:00:00	104.00	104.00	104.00	104.00	104.00	104.000000	104.000000	104.00	104.00000
9	2003-03-01 01:00:00	104.00	1.92	104.00	104.00	104.00	71.839996	181.399994	104.00	5.33000

In [7]:

```
d=c[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
 'PM10', 'PXY', 'SO_2', 'TCH', 'TOL', 'station']]
d
```

Out[7]:

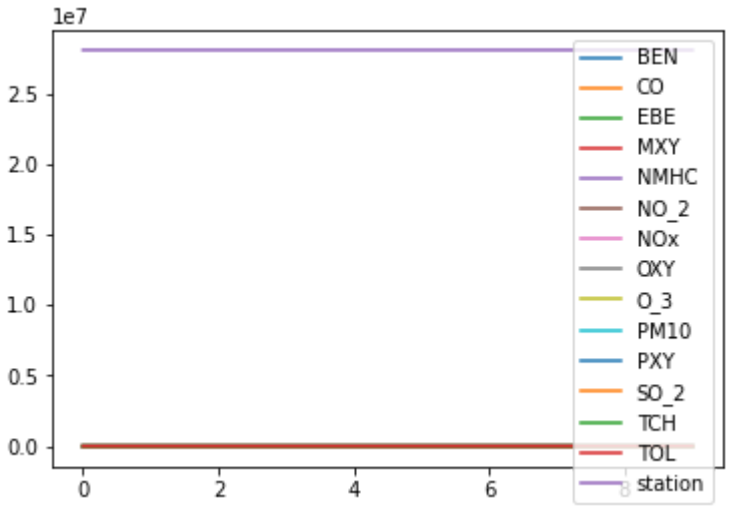
	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	P
0	104.00	1.72	104.00	104.00	104.00	73.900002	316.299988	104.00	10.550000	55.20
1	104.00	1.45	104.00	104.00	0.26	72.110001	250.000000	0.73	6.720000	52.38
2	104.00	1.57	104.00	104.00	104.00	80.559998	224.199997	104.00	21.049999	63.24
3	104.00	2.45	104.00	104.00	104.00	78.370003	450.399994	104.00	4.220000	67.83
4	104.00	3.26	104.00	104.00	104.00	96.250000	479.100006	104.00	8.460000	95.77
5	8.41	1.94	9.83	21.49	0.45	90.300003	384.899994	9.48	9.950000	95.15
6	104.00	1.38	104.00	104.00	0.29	89.580002	230.000000	104.00	7.200000	54.00
7	104.00	1.58	104.00	104.00	0.30	93.639999	334.600006	104.00	4.190000	26.62
8	104.00	104.00	104.00	104.00	104.00	104.000000	104.000000	104.00	104.000000	104.00
9	104.00	1.92	104.00	104.00	104.00	71.839996	181.399994	104.00	5.330000	39.36

In [8]:

```
d.plot.line()
```

Out[8]:

```
<AxesSubplot:>
```

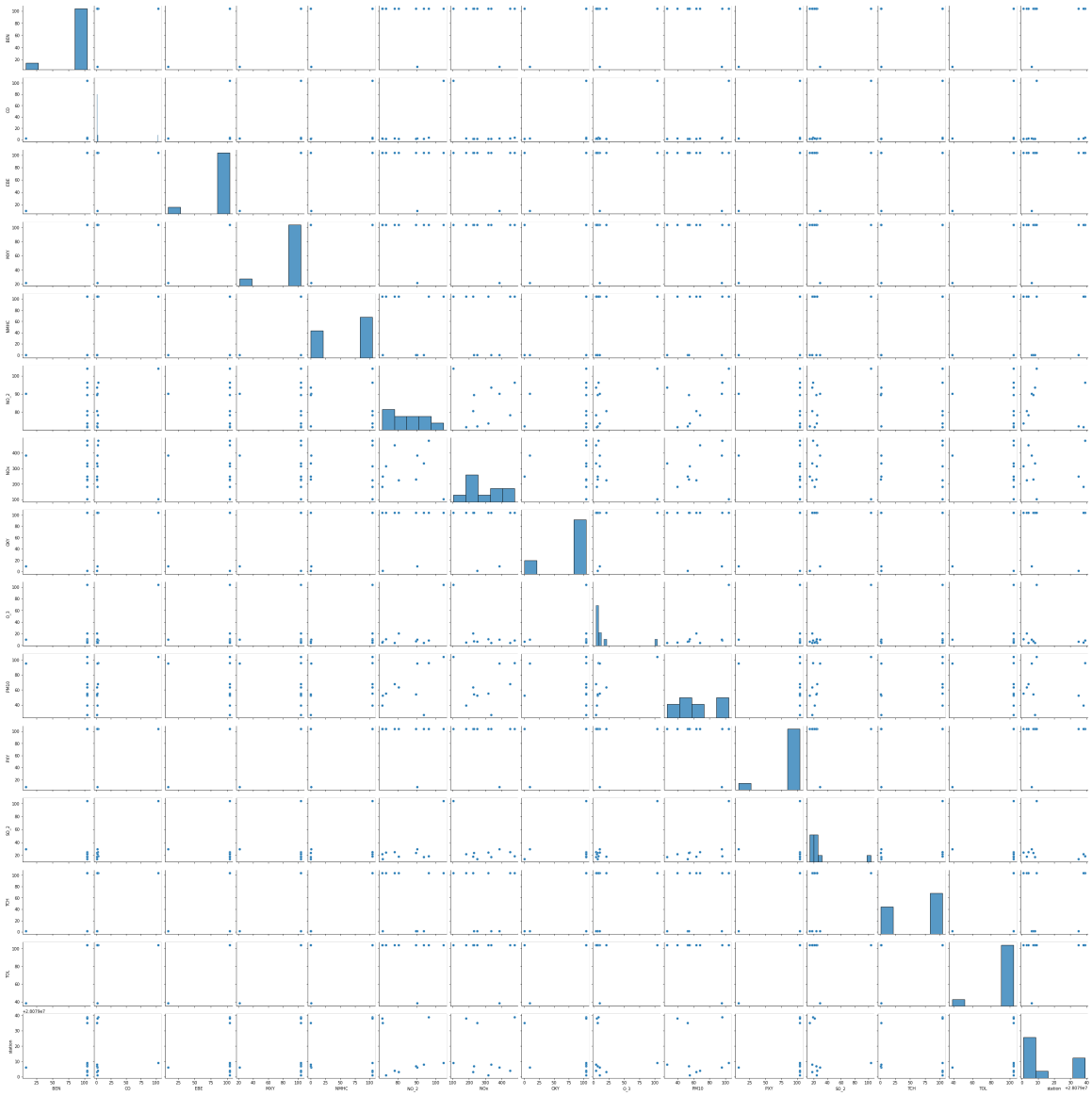


In [9]:

```
sns.pairplot(d)
```

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x1179c4877c0>
```



In [10]:

```
x=d[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY']]
y=d['TCH']
```

In [11]:

```
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)
```

In [12]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[12]:

```
LinearRegression()
```

In [13]:

```
print(lr.intercept_)
```

```
1.1450565709301515
```

In [14]:

```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[14]:

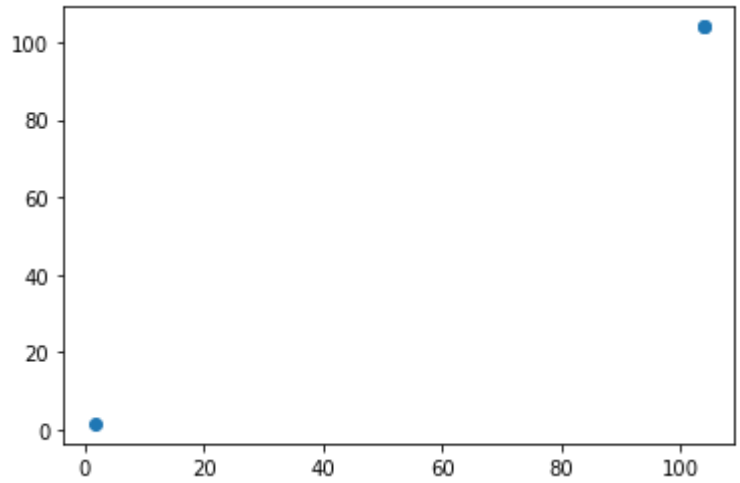
	Co-efficient
BEN	5.015980e-04
CO	-1.410830e-15
EBE	4.941467e-04
MXY	4.329622e-04
NMHC	9.884293e-01
NO_2	2.725859e-16
NOx	-3.231354e-16
OXY	-8.681400e-04

In [15]:

```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:

```
<matplotlib.collections.PathCollection at 0x117af35d400>
```



In [16]:

```
print(lr.score(x_test,y_test))
```

```
0.9999990821806635
```

In [17]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [18]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[18]:

```
Ridge(alpha=10)
```

In [19]:

```
rr.score(x_test,y_test)
```

Out[19]:

```
0.9999407608577562
```

In [20]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[20]:

```
Lasso(alpha=10)
```

In [21]:

```
la.score(x_test,y_test)
```

Out[21]:

```
0.9999882336568255
```

In [22]:

```
a1=b.head(7000)
a1
```

Out[22]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	
0	2003-03-01 01:00:00	104.00	1.72	104.00	104.00	104.00	73.900002	316.299988	104.00	10.55
1	2003-03-01 01:00:00	104.00	1.45	104.00	104.00	0.26	72.110001	250.000000	0.73	6.72
2	2003-03-01 01:00:00	104.00	1.57	104.00	104.00	104.00	80.559998	224.199997	104.00	21.04
3	2003-03-01 01:00:00	104.00	2.45	104.00	104.00	104.00	78.370003	450.399994	104.00	4.22
4	2003-03-01 01:00:00	104.00	3.26	104.00	104.00	104.00	96.250000	479.100006	104.00	8.46
...	...	...	...	...	...	...	...	...	...	
6995	2003-03-11 10:00:00	1.53	0.88	1.50	2.96	0.17	51.119999	154.800003	1.42	8.69
6996	2003-03-11 10:00:00	3.68	0.81	3.72	8.24	104.00	143.300003	408.799988	0.59	5.86
6997	2003-03-11 10:00:00	104.00	104.00	104.00	104.00	0.22	108.199997	305.000000	104.00	12.92
6998	2003-03-11 10:00:00	104.00	104.00	104.00	104.00	0.13	95.540001	292.500000	104.00	104.00
6999	2003-03-11 10:00:00	4.21	1.75	2.81	8.05	0.26	96.910004	289.000000	2.45	9.69

7000 rows × 16 columns

In [23]:

```
e=a1[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
 'PM10', 'PXY', 'SO_2', 'TCH', 'TOL', 'station']]
```

In [24]:

```
f=e.iloc[:,0:14]
g=e.iloc[:,-1]
```

In [25]:

```
h=StandardScaler().fit_transform(f)
```

In [26]:

```
logr=LogisticRegression(max_iter=10000)
logr.fit(h,g)
```

Out[26]:

```
LogisticRegression(max_iter=10000)
```

In [27]:

```
from sklearn.model_selection import train_test_split
h_train,h_test,g_train,g_test=train_test_split(h,g,test_size=0.3)
```

In [28]:

```
i=[[10,20,30,40,50,60,11,22,33,44,55,54,21,78]]
```

In [29]:

```
prediction=logr.predict(i)
print(prediction)
```

```
[28079003]
```

In [30]:

```
logr.classes_
```

Out[30]:

```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079017, 28079018, 28079019, 28079021, 28079022, 28079023,
       28079024, 28079025, 28079026, 28079027, 28079035, 28079036,
       28079038, 28079039, 28079040, 28079099], dtype=int64)
```

In [31]:

```
logr.predict_proba(i)[0][0]
```

Out[31]:

```
2.4613648777748206e-09
```

In [32]:

```
logr.predict_proba(i)[0][1]
```

Out[32]:

```
0.999999899172892
```

In [33]:

```
logr.score(h_test,g_test)
```

Out[33]:

```
0.5947619047619047
```

In [34]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[34]:

```
ElasticNet()
```

In [35]:

```
print(en.coef_)
```

```
[ 6.12161669e-06  0.00000000e+00  7.44059496e-04  0.00000000e+00
  9.87690805e-01 -0.00000000e+00 -0.00000000e+00  0.00000000e+00]
```

In [36]:

```
print(en.intercept_)
```

```
1.1853508091741816
```

In [37]:

```
prediction=en.predict(x_test)
print(en.score(x_test,y_test))
```

```
0.9999998603491785
```

In [38]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(h_train,g_train)
```

Out[38]:

```
RandomForestClassifier()
```

In [39]:

```
parameters={'max_depth':[1,2,3,4,5],
 'min_samples_leaf':[5,10,15,20,25],
 'n_estimators':[10,20,30,40,50]
 }
```

In [40]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(h_train,g_train)
```

Out[40]:

```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [41]:

```
grid_search.best_score_
```

Out[41]:

```
0.5708163265306122
```

In [42]:

```
rfc_best=grid_search.best_estimator_
```

In [43]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,50))
plot_tree(rfc_best.estimators_[2],filled=True)
```

```
 Text(202.9090909090909, 679.5, 'X[1] <= -0.296\ngini = 0.447\nsamples =
115\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 128,
0, 0, 0, 21, 0, 0, 0, 0, 30]'),
 Text(101.45454545454545, 226.5, 'gini = 0.589\nsamples = 56\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 42, 0, 0, 0, 10, 0, 0,
0, 0, 30]'),
 Text(304.3636363636364, 226.5, 'gini = 0.201\nsamples = 59\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 86, 0, 0, 0, 11, 0, 0,
0, 0, 0]'),
 Text(608.7272727272727, 679.5, 'X[0] <= -1.655\ngini = 0.758\nsamples =
213\nvalue = [0, 0, 0, 0, 16, 24, 0, 0, 0, 0, 22, 0, 0, 59\n0, 0, 0, 85,
0, 0, 0, 0, 6, 0, 0, 0, 123, 0]'),
 Text(507.27272727272725, 226.5, 'gini = 0.0\nsamples = 52\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 84, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(710.1818181818181, 226.5, 'gini = 0.683\nsamples = 161\nvalue = [0,
0, 0, 0, 16, 24, 0, 0, 0, 0, 22, 0, 0, 59\n0, 0, 0, 1, 0, 0, 0, 0, 6, 0,
0, 0, 123, 0]'),
 Text(1217.4545454545455, 1132.5, 'X[6] <= -0.735\ngini = 0.491\nsamples
= 186\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0
```

In [ ]: