In [130]:

```
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 [173]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2005.csv")
a
```

Out[173]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM
0	2005-11-01 01:00:00	NaN	0.77	NaN	NaN	NaN	57.130001	128.699997	NaN	14.720000	14.
1	2005-11-01 01:00:00	1.52	0.65	1.49	4.57	0.25	86.559998	181.699997	1.27	11.680000	30.
2	2005-11-01 01:00:00	NaN	0.40	NaN	NaN	NaN	46.119999	53.000000	NaN	30.469999	14.
3	2005-11-01 01:00:00	NaN	0.42	NaN	NaN	NaN	37.220001	52.009998	NaN	21.379999	15.
4	2005-11-01 01:00:00	NaN	0.57	NaN	NaN	NaN	32.160000	36.680000	NaN	33.410000	5.
...	...	...	...	...	...	...	...	...	...	...	
236995	2006-01-01 00:00:00	1.08	0.36	1.01	NaN	0.11	21.990000	23.610001	NaN	43.349998	5.
236996	2006-01-01 00:00:00	0.39	0.54	1.00	1.00	0.11	2.200000	4.220000	1.00	69.639999	4.
236997	2006-01-01 00:00:00	0.19	NaN	0.26	NaN	0.08	26.730000	30.809999	NaN	43.840000	4.
236998	2006-01-01 00:00:00	0.14	NaN	1.00	NaN	0.06	13.770000	17.770000	NaN	NaN	5.
236999	2006-01-01 00:00:00	0.50	0.40	0.73	1.84	0.13	20.940001	26.950001	1.49	48.259998	5.

237000 rows × 17 columns

In [174]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 237000 entries, 0 to 236999
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     237000 non-null  object 
 1   BEN      70370 non-null   float64
 2   CO       217656 non-null  float64
 3   EBE      68955 non-null   float64
 4   MXY      32549 non-null   float64
 5   NMHC     92854 non-null   float64
 6   NO_2     235022 non-null  float64
 7   NOx      235049 non-null  float64
 8   OXY      32555 non-null   float64
 9   O_3      223162 non-null  float64
 10  PM10     232142 non-null  float64
 11  PM25     69407 non-null   float64
 12  PXY      32549 non-null   float64
 13  SO_2     235277 non-null  float64
 14  TCH      93076 non-null   float64
 15  TOL      70255 non-null   float64
 16  station  237000 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 30.7+ MB
```

In [175]:

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

Out[175]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	
0	2005-11-01 01:00:00	104.00	0.77	104.00	104.00	104.00	57.130001	128.699997	104.00	14.7
1	2005-11-01 01:00:00	1.52	0.65	1.49	4.57	0.25	86.559998	181.699997	1.27	11.6
2	2005-11-01 01:00:00	104.00	0.40	104.00	104.00	104.00	46.119999	53.000000	104.00	30.4
3	2005-11-01 01:00:00	104.00	0.42	104.00	104.00	104.00	37.220001	52.009998	104.00	21.3
4	2005-11-01 01:00:00	104.00	0.57	104.00	104.00	104.00	32.160000	36.680000	104.00	33.4
...	...	...	...	...	...	...	...	...	...	
236995	2006-01-01 00:00:00	1.08	0.36	1.01	104.00	0.11	21.990000	23.610001	104.00	43.3
236996	2006-01-01 00:00:00	0.39	0.54	1.00	1.00	0.11	2.200000	4.220000	1.00	69.6
236997	2006-01-01 00:00:00	0.19	104.00	0.26	104.00	0.08	26.730000	30.809999	104.00	43.8
236998	2006-01-01 00:00:00	0.14	104.00	1.00	104.00	0.06	13.770000	17.770000	104.00	104.0
236999	2006-01-01 00:00:00	0.50	0.40	0.73	1.84	0.13	20.940001	26.950001	1.49	48.2

237000 rows × 17 columns

In [176]:

```
b.columns
```

Out[176]:

```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O
_3',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [177]:

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

Out[177]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2005-11-01 01:00:00	104.00	0.77	104.00	104.00	104.00	57.130001	128.699997	104.00	14.720000	1
1	2005-11-01 01:00:00	1.52	0.65	1.49	4.57	0.25	86.559998	181.699997	1.27	11.680000	3
2	2005-11-01 01:00:00	104.00	0.40	104.00	104.00	104.00	46.119999	53.000000	104.00	30.469999	1
3	2005-11-01 01:00:00	104.00	0.42	104.00	104.00	104.00	37.220001	52.009998	104.00	21.379999	1
4	2005-11-01 01:00:00	104.00	0.57	104.00	104.00	104.00	32.160000	36.680000	104.00	33.410000	
5	2005-11-01 01:00:00	1.92	0.88	2.44	5.14	0.22	90.309998	207.699997	2.78	13.760000	1
6	2005-11-01 01:00:00	104.00	0.55	104.00	104.00	0.27	50.279999	77.209999	104.00	19.120001	1
7	2005-11-01 01:00:00	0.20	0.38	1.00	104.00	0.27	51.759998	72.989998	104.00	14.810000	1
8	2005-11-01 01:00:00	104.00	0.70	104.00	104.00	104.00	39.040001	43.860001	104.00	25.379999	1
9	2005-11-01 01:00:00	104.00	0.56	104.00	104.00	104.00	41.820000	51.869999	104.00	24.290001	

In [178]:

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

Out[178]:

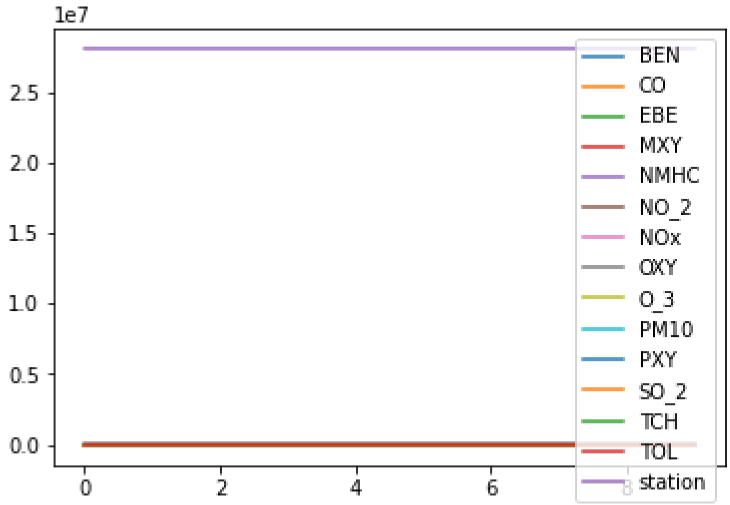
	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	104.00	0.77	104.00	104.00	104.00	57.130001	128.699997	104.00	14.720000	14.910000
1	1.52	0.65	1.49	4.57	0.25	86.559998	181.699997	1.27	11.680000	30.930000
2	104.00	0.40	104.00	104.00	104.00	46.119999	53.000000	104.00	30.469999	14.600000
3	104.00	0.42	104.00	104.00	104.00	37.220001	52.009998	104.00	21.379999	15.160000
4	104.00	0.57	104.00	104.00	104.00	32.160000	36.680000	104.00	33.410000	5.000000
5	1.92	0.88	2.44	5.14	0.22	90.309998	207.699997	2.78	13.760000	18.070000
6	104.00	0.55	104.00	104.00	0.27	50.279999	77.209999	104.00	19.120001	18.209999
7	0.20	0.38	1.00	104.00	0.27	51.759998	72.989998	104.00	14.810000	16.430000
8	104.00	0.70	104.00	104.00	104.00	39.040001	43.860001	104.00	25.379999	16.139999
9	104.00	0.56	104.00	104.00	104.00	41.820000	51.869999	104.00	24.290001	7.130000

In [179]:

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

Out[179]:

```
<AxesSubplot:>
```

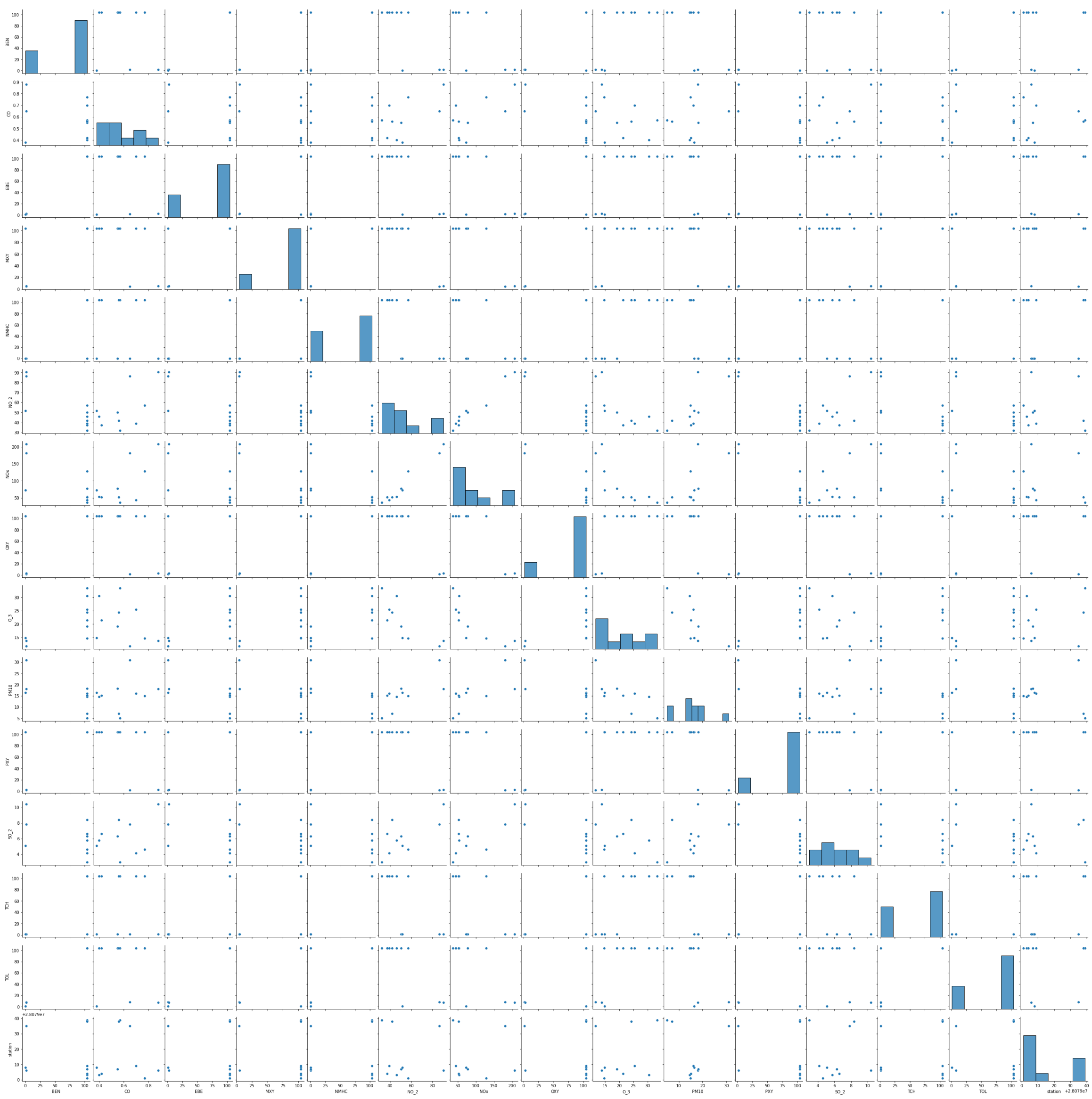


In [180]:

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

Out[180]:

```
<seaborn.axisgrid.PairGrid at 0x117f4c8e4c0>
```



In [181]:

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

In [182]:

```
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 [183]:

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

Out[183]:

```
LinearRegression()
```

In [184]:

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

```
1.303925233642957
```

In [185]:

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

Out[185]:

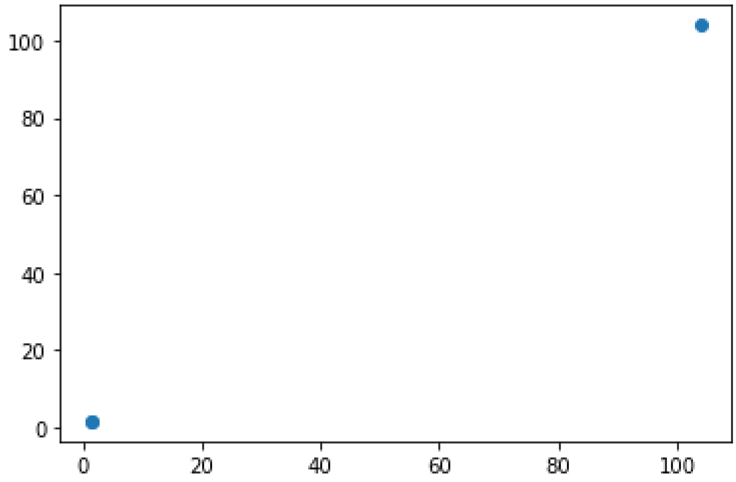
	Co-efficient
BEN	-8.252184e-04
CO	3.009027e-14
EBE	-8.188583e-04
MXY	0.000000e+00
NMHC	9.891063e-01
NO_2	-1.841709e-15
NOx	2.687275e-16
OXY	0.000000e+00

In [186]:

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

Out[186]:

```
<matplotlib.collections.PathCollection at 0x11784320c40>
```



In [187]:

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

```
0.9999916666941001
```

In [188]:

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

In [189]:

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

Out[189]:

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

In [190]:

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

Out[190]:

```
0.9999986140942656
```

In [191]:

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

Out[191]:

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

In [192]:

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

Out[192]:

```
0.999946200042451
```

In [193]:

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

Out[193]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	
0	2005-11-01 01:00:00	104.00	0.77	104.00	104.00	104.00	57.130001	128.699997	104.00	14.72
1	2005-11-01 01:00:00	1.52	0.65	1.49	4.57	0.25	86.559998	181.699997	1.27	11.68
2	2005-11-01 01:00:00	104.00	0.40	104.00	104.00	104.00	46.119999	53.000000	104.00	30.46
3	2005-11-01 01:00:00	104.00	0.42	104.00	104.00	104.00	37.220001	52.009998	104.00	21.37
4	2005-11-01 01:00:00	104.00	0.57	104.00	104.00	104.00	32.160000	36.680000	104.00	33.41
...	...	...	...	...	...	...	...	...	...	
6995	2005-11-11 21:00:00	1.11	0.56	1.85	4.41	0.25	73.570000	100.599998	1.33	11.45
6996	2005-11-11 21:00:00	0.49	104.00	0.25	104.00	0.14	119.800003	254.500000	104.00	2.06
6997	2005-11-11 21:00:00	0.25	104.00	0.51	104.00	0.10	73.500000	104.300003	104.00	104.00
6998	2005-11-11 21:00:00	1.59	0.83	2.06	8.59	0.26	87.279999	118.400002	3.23	7.39
6999	2005-11-11 22:00:00	104.00	0.78	104.00	104.00	104.00	53.900002	166.000000	104.00	11.82

7000 rows × 17 columns

In [194]:

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

In [195]:

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

In [196]:

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

In [197]:

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

Out[197]:

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

In [198]:

```
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 [199]:

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

In [200]:

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

```
[28079039]
```

In [201]:

```
logr.classes_
```

Out[201]:

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

In [202]:

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

Out[202]:

```
4.2610216372313485e-271
```

In [203]:

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

Out[203]:

```
1.3745976634115662e-198
```

In [204]:

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

Out[204]:

```
0.5223809523809524
```

In [205]:

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

Out[205]:

```
ElasticNet()
```

In [206]:

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

```
[-3.88704777e-04 -0.00000000e+00 -0.00000000e+00  0.00000000e+00
  9.88028714e-01 -0.00000000e+00 -0.00000000e+00  0.00000000e+00]
```

In [207]:

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

```
1.2720232497242563
```

In [208]:

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

```
0.9999942535439239
```

In [209]:

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

Out[209]:

```
RandomForestClassifier()
```

In [210]:

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

In [211]:

```
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[211]:

```
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 [212]:

```
grid_search.best_score_
```

Out[212]:

```
0.5463265306122449
```

In [213]:

```
rfc_best=grid_search.best_estimator_
```

In [214]:

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

Out[214]:

```
[Text(2148.3, 2491.5, 'X[7] <= -0.528\ngini = 0.962\nsamples = 3106\nvalu
e = [192, 184, 163, 221, 174, 189, 170, 189, 190, 148\n214, 181, 172, 21
8, 178, 189, 212, 182, 168, 161\n209, 181, 139, 170, 158, 163, 185]'),
 Text(1023.0000000000001, 2038.5, 'X[13] <= -1.505\ngini = 0.744\nsamples
= 456\nvalue = [0, 0, 0, 211, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 1
37, 0, 0, 181, 0, 0, 0, 0, 185]'),
 Text(558.0, 1585.5, 'X[10] <= -2.528\ngini = 0.704\nsamples = 165\nvalue
= [0, 0, 0, 26, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 97, 0, 0, 51,
0, 0, 0, 0, 79]'),
 Text(297.6, 1132.5, 'X[11] <= -0.556\ngini = 0.628\nsamples = 120\nvalue
= [0, 0, 0, 10, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 96, 0, 0, 50,
0, 0, 0, 0, 28]'),
 Text(148.8, 679.5, 'X[8] <= 0.555\ngini = 0.18\nsamples = 14\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 2, 0, 0, 18, 0, 0,
0, 0, 0]'),
 Text(74.4, 226.5, 'gini = 0.0\nsamples = 9\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 15, 0, 0, 0, 0, 0]'),
 Text(223.20000000000002, 226.5, 'gini = 0.48\nsamples = 5\nvalue = [0,
```

In [ ]:

```

```