In [130]:

```python
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 [215]:

```python
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2006.csv")
a
```

Out[215]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2006-02-01 01:00:00	NaN	1.84	NaN	NaN	NaN	155.100006	490.100006	NaN	4.880000	97
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.36	0.32	94.339996	229.699997	3.04	7.100000	25
2	2006-02-01 01:00:00	NaN	1.25	NaN	NaN	NaN	66.800003	192.000000	NaN	4.430000	34
3	2006-02-01 01:00:00	NaN	1.68	NaN	NaN	NaN	103.000000	407.799988	NaN	4.830000	28
4	2006-02-01 01:00:00	NaN	1.31	NaN	NaN	NaN	105.400002	269.200012	NaN	6.990000	54
...	...	...	...	...	...	...	...	...	...	...	
230563	2006-05-01 00:00:00	5.88	0.83	6.23	NaN	0.20	112.500000	218.000000	NaN	24.389999	93
230564	2006-05-01 00:00:00	0.76	0.32	0.48	1.09	0.08	51.900002	54.820000	0.61	48.410000	29
230565	2006-05-01 00:00:00	0.96	NaN	0.69	NaN	0.19	135.100006	179.199997	NaN	11.460000	64
230566	2006-05-01 00:00:00	0.50	NaN	0.67	NaN	0.10	82.599998	105.599998	NaN	NaN	94
230567	2006-05-01 00:00:00	1.95	0.74	1.99	4.00	0.24	107.300003	160.199997	2.01	17.730000	52

230568 rows × 17 columns

In [216]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 230568 entries, 0 to 230567
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     230568 non-null  object 
 1   BEN      73979 non-null   float64
 2   CO       211665 non-null  float64
 3   EBE      73948 non-null   float64
 4   MXY      33422 non-null   float64
 5   NMHC     90829 non-null   float64
 6   NO_2     228855 non-null  float64
 7   NOx      228855 non-null  float64
 8   OXY      33472 non-null   float64
 9   O_3      216511 non-null  float64
 10  PM10     227469 non-null  float64
 11  PM25     61758 non-null   float64
 12  PXY      33447 non-null   float64
 13  SO_2     229125 non-null  float64
 14  TCH      90887 non-null   float64
 15  TOL      73840 non-null   float64
 16  station  230568 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 29.9+ MB
```

In [217]:

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

Out[217]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	
0	2006-02-01 01:00:00	104.00	1.84	104.00	104.00	104.00	155.100006	490.100006	104.00	4.
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.36	0.32	94.339996	229.699997	3.04	7.
2	2006-02-01 01:00:00	104.00	1.25	104.00	104.00	104.00	66.800003	192.000000	104.00	4.
3	2006-02-01 01:00:00	104.00	1.68	104.00	104.00	104.00	103.000000	407.799988	104.00	4.
4	2006-02-01 01:00:00	104.00	1.31	104.00	104.00	104.00	105.400002	269.200012	104.00	6.
...	...	...	...	...	...	...	...	...	...	
230563	2006-05-01 00:00:00	5.88	0.83	6.23	104.00	0.20	112.500000	218.000000	104.00	24.
230564	2006-05-01 00:00:00	0.76	0.32	0.48	1.09	0.08	51.900002	54.820000	0.61	48.
230565	2006-05-01 00:00:00	0.96	104.00	0.69	104.00	0.19	135.100006	179.199997	104.00	11.
230566	2006-05-01 00:00:00	0.50	104.00	0.67	104.00	0.10	82.599998	105.599998	104.00	104.
230567	2006-05-01 00:00:00	1.95	0.74	1.99	4.00	0.24	107.300003	160.199997	2.01	17.

230568 rows × 17 columns

In [218]:

```
b.columns
```

Out[218]:

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

In [219]:

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

Out[219]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	2006-02-01 01:00:00	104.00	1.84	104.00	104.000000	104.00	155.100006	490.100006	104.00	4.88
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.360000	0.32	94.339996	229.699997	3.04	7.10
2	2006-02-01 01:00:00	104.00	1.25	104.00	104.000000	104.00	66.800003	192.000000	104.00	4.43
3	2006-02-01 01:00:00	104.00	1.68	104.00	104.000000	104.00	103.000000	407.799988	104.00	4.83
4	2006-02-01 01:00:00	104.00	1.31	104.00	104.000000	104.00	105.400002	269.200012	104.00	6.99
5	2006-02-01 01:00:00	9.41	1.69	9.98	19.959999	0.44	142.199997	453.500000	11.31	5.99
6	2006-02-01 01:00:00	104.00	1.28	104.00	104.000000	0.57	94.320000	294.000000	104.00	6.77
7	2006-02-01 01:00:00	0.27	1.51	0.28	104.000000	0.46	144.699997	385.299988	104.00	5.30
8	2006-02-01 01:00:00	104.00	2.65	104.00	104.000000	104.00	197.100006	673.099976	104.00	2.64
9	2006-02-01 01:00:00	104.00	1.30	104.00	104.000000	104.00	130.899994	282.000000	104.00	5.14

In [220]:

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

Out[220]:

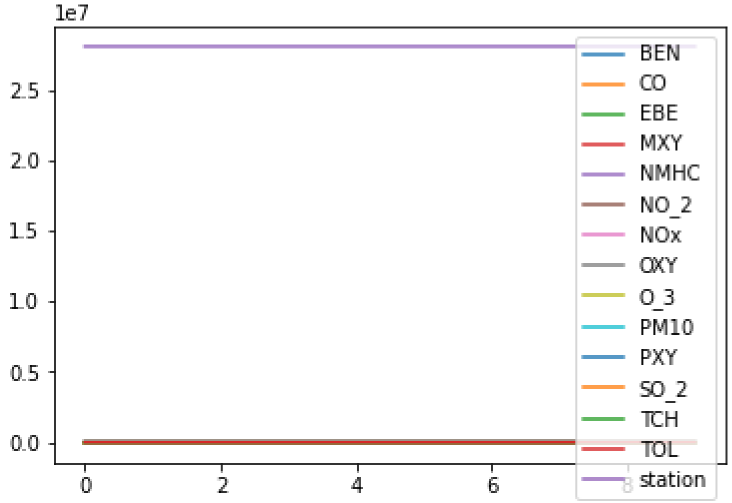
	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	104.00	1.84	104.00	104.000000	104.00	155.100006	490.100006	104.00	4.88	97.570000
1	1.68	1.01	2.38	6.360000	0.32	94.339996	229.699997	3.04	7.10	25.820000
2	104.00	1.25	104.00	104.000000	104.00	66.800003	192.000000	104.00	4.43	34.419998
3	104.00	1.68	104.00	104.000000	104.00	103.000000	407.799988	104.00	4.83	28.260000
4	104.00	1.31	104.00	104.000000	104.00	105.400002	269.200012	104.00	6.99	54.180000
5	9.41	1.69	9.98	19.959999	0.44	142.199997	453.500000	11.31	5.99	89.190002
6	104.00	1.28	104.00	104.000000	0.57	94.320000	294.000000	104.00	6.77	55.130001
7	0.27	1.51	0.28	104.000000	0.46	144.699997	385.299988	104.00	5.30	80.150002
8	104.00	2.65	104.00	104.000000	104.00	197.100006	673.099976	104.00	2.64	142.500000
9	104.00	1.30	104.00	104.000000	104.00	130.899994	282.000000	104.00	5.14	49.029999

In [221]:

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

Out[221]:

<AxesSubplot:>

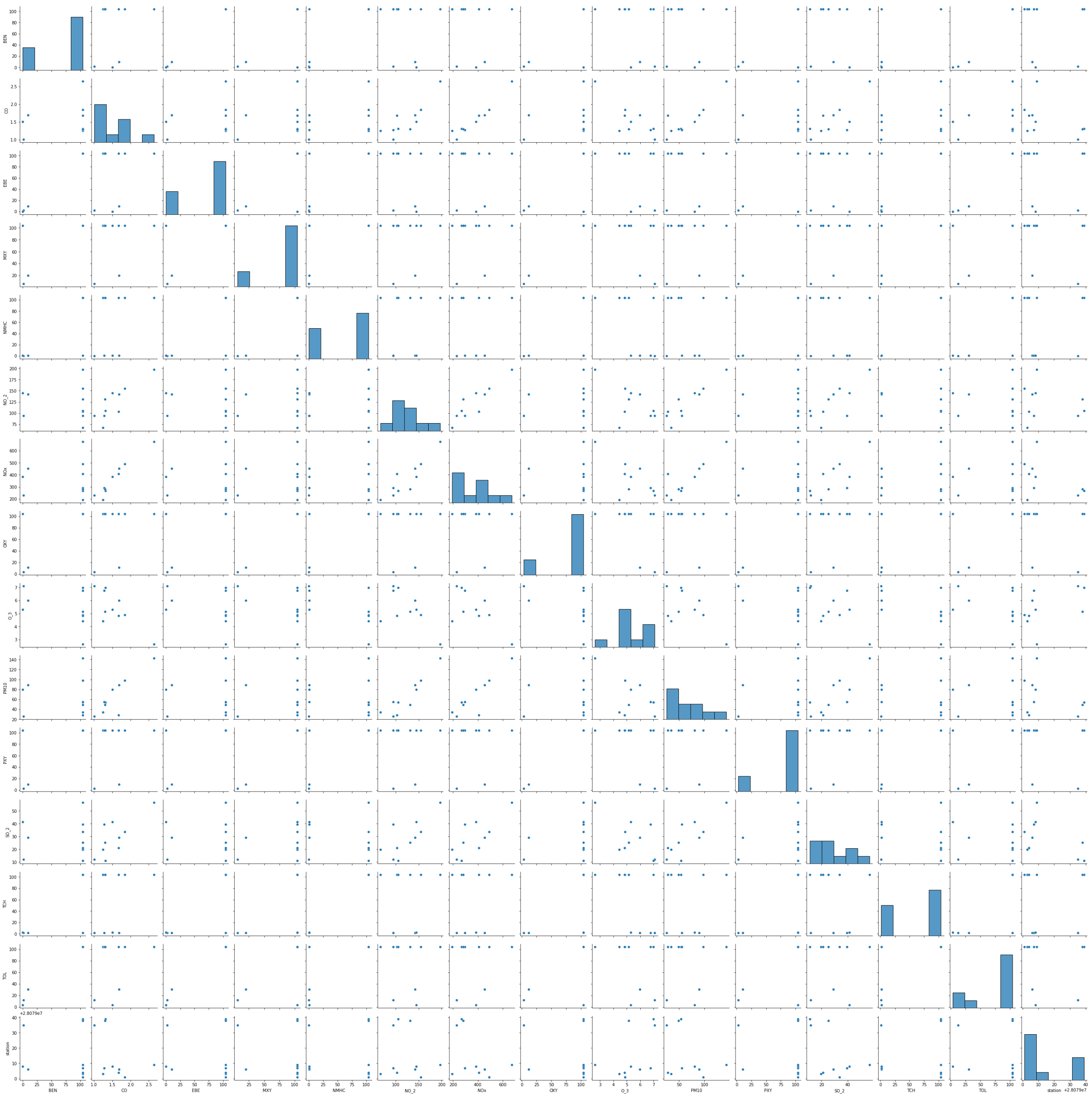


In [222]:

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

Out[222]:

```
<seaborn.axisgrid.PairGrid at 0x1178a76efa0>
```



In [223]:

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

In [224]:

```
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 [225]:

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

Out[225]:

```
LinearRegression()
```

In [226]:

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

```
1.6366528371229805
```

In [227]:

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

Out[227]:

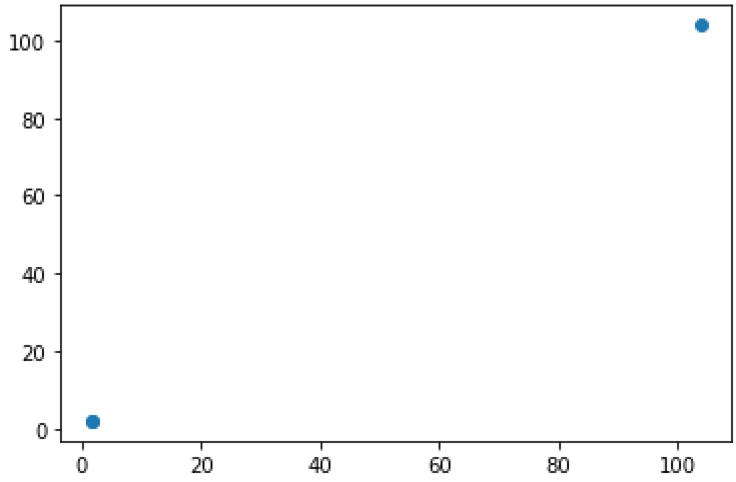
	Co-efficient
BEN	-1.584156e-03
CO	-8.575021e-15
EBE	-1.584003e-03
MXY	0.000000e+00
NMHC	9.874311e-01
NO_2	-4.906496e-16
NOx	-5.630549e-18
OXY	0.000000e+00

In [228]:

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

Out[228]:

```
<matplotlib.collections.PathCollection at 0x117972360d0>
```



In [229]:

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

```
0.9999771384355651
```

In [230]:

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

In [231]:

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

Out[231]:

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

In [232]:

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

Out[232]:

```
0.999977789448488
```

In [233]:

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

Out[233]:

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

In [234]:

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

Out[234]:

```
0.9999348061232213
```

In [235]:

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

Out[235]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2006-02-01 01:00:00	104.00	1.84	104.00	104.00	104.00	155.100006	490.100006	104.00	4.88	9
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.36	0.32	94.339996	229.699997	3.04	7.10	2
2	2006-02-01 01:00:00	104.00	1.25	104.00	104.00	104.00	66.800003	192.000000	104.00	4.43	3
3	2006-02-01 01:00:00	104.00	1.68	104.00	104.00	104.00	103.000000	407.799988	104.00	4.83	2
4	2006-02-01 01:00:00	104.00	1.31	104.00	104.00	104.00	105.400002	269.200012	104.00	6.99	5
...	...	...	...	...	...	...	...	...	...	...	
6995	2006-02-12 06:00:00	1.54	0.44	2.80	7.86	0.19	61.410000	84.349998	2.85	8.77	1
6996	2006-02-12 06:00:00	104.00	0.46	104.00	104.00	104.00	53.340000	75.160004	104.00	7.88	1
6997	2006-02-12 06:00:00	104.00	1.06	104.00	104.00	104.00	73.279999	231.899994	104.00	4.38	2
6998	2006-02-12 06:00:00	104.00	0.57	104.00	104.00	104.00	47.400002	52.240002	104.00	15.17	2
6999	2006-02-12 06:00:00	2.12	0.66	2.39	4.76	0.11	74.879997	163.600006	2.69	8.16	2

7000 rows × 17 columns

In [236]:

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

In [237]:

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

In [238]:

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

In [239]:

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

Out[239]:

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

In [240]:

```
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 [241]:

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

In [242]:

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

```
[28079039]
```

In [243]:

```
logr.classes_
```

Out[243]:

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

In [244]:

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

Out[244]:

```
1.9164523757870783e-126
```

In [245]:

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

Out[245]:

```
1.0553514949951003e-174
```

In [246]:

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

Out[246]:

```
0.5476190476190477
```

In [247]:

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

Out[247]:

```
ElasticNet()
```

In [248]:

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

```
[-1.76568773e-03  0.00000000e+00 -5.13577003e-06  0.00000000e+00
  9.86248953e-01  0.00000000e+00  5.12700660e-05  0.00000000e+00]
```

In [249]:

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

```
1.5801659139574014
```

In [250]:

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

```
0.9999820518825869
```

In [251]:

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

Out[251]:

```
RandomForestClassifier()
```

In [252]:

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

In [253]:

```
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[253]:

```
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 [254]:

```
grid_search.best_score_
```

Out[254]:

```
0.5751020408163265
```

In [255]:

```
rfc_best=grid_search.best_estimator_
```

In [256]:

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

```
e = [167, 195, 214, 179, 176, 178, 173, 179, 187, 187\n176, 187, 179, 18
0, 180, 229, 204, 177, 200, 186\n184, 196, 188, 196, 193, 210]'),
 Text(1249.92, 2038.5, 'X[2] <= -1.335\ngini = 0.749\nsamples = 482\nvalu
e = [0, 0, 0, 179, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 177, 0, 0, 184,
0, 0, 0, 0, 210]'),
 Text(714.24, 1585.5, 'X[10] <= -2.371\ngini = 0.71\nsamples = 351\nvalue
= [0, 0, 0, 51, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 170, 0, 0, 127, 0,
0, 0, 0, 192]'),
 Text(357.12, 1132.5, 'X[4] <= -1.175\ngini = 0.626\nsamples = 191\nvalue
= [0, 0, 0, 30, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 165, 0, 0, 55, 0,
0, 0, 0, 50]'),
 Text(178.56, 679.5, 'X[12] <= -1.177\ngini = 0.49\nsamples = 88\nvalue =
[0, 0, 0, 23, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 94, 0, 0, 0, 0, 0,
0, 0, 22]'),
 Text(89.28, 226.5, 'gini = 0.408\nsamples = 12\nvalue = [0, 0, 0, 15, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(267.84000000000003, 226.5, 'gini = 0.404\nsamples = 76\nvalue = [0,
0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 88, 0, 0, 0, 0, 0, 0, 0,
22]'),
 Text(535.6800000000001, 679.5, 'X[9] <= -0.907\ngini = 0.657\nsamples =
```

In [ ]:

```

```