In [44]:

```
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 [45]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2001.csv")
a
```

Out[45]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2001-08-01 01:00:00	NaN	0.37	NaN	NaN	NaN	58.400002	87.150002	NaN	34.529999	10
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	10
2	2001-08-01 01:00:00	NaN	0.28	NaN	NaN	NaN	50.660000	61.380001	NaN	46.310001	10
3	2001-08-01 01:00:00	NaN	0.47	NaN	NaN	NaN	69.790001	73.449997	NaN	40.650002	6
4	2001-08-01 01:00:00	NaN	0.39	NaN	NaN	NaN	22.830000	24.799999	NaN	66.309998	7
...	...	...	...	...	...	...	...	...	...	...	
217867	2001-04-01 00:00:00	10.45	1.81	NaN	NaN	NaN	73.000000	264.399994	NaN	5.200000	4
217868	2001-04-01 00:00:00	5.20	0.69	4.56	NaN	0.13	71.080002	129.300003	NaN	13.460000	2
217869	2001-04-01 00:00:00	0.49	1.09	NaN	1.00	0.19	76.279999	128.399994	0.35	5.020000	4
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.38	NaN	80.019997	197.000000	2.58	5.840000	3
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.04	0.18	76.809998	206.300003	5.20	8.340000	3

217872 rows × 16 columns

In [46]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 217872 entries, 0 to 217871
Data columns (total 16 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     217872 non-null  object 
 1   BEN      70389 non-null   float64
 2   CO       216341 non-null  float64
 3   EBE      57752 non-null   float64
 4   MXY      42753 non-null   float64
 5   NMHC     85719 non-null   float64
 6   NO_2     216331 non-null  float64
 7   NOx      216318 non-null  float64
 8   OXY      42856 non-null   float64
 9   O_3      216514 non-null  float64
 10  PM10     207776 non-null  float64
 11  PXY      42845 non-null   float64
 12  SO_2     216403 non-null  float64
 13  TCH      85797 non-null   float64
 14  TOL      70196 non-null   float64
 15  station  217872 non-null  int64  
dtypes: float64(14), int64(1), object(1)
memory usage: 26.6+ MB
```

In [47]:

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

Out[47]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O
0	2001-08-01 01:00:00	104.00	0.37	104.00	104.00	104.00	58.400002	87.150002	104.00	34.5299
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.1600
2	2001-08-01 01:00:00	104.00	0.28	104.00	104.00	104.00	50.660000	61.380001	104.00	46.3100
3	2001-08-01 01:00:00	104.00	0.47	104.00	104.00	104.00	69.790001	73.449997	104.00	40.6500
4	2001-08-01 01:00:00	104.00	0.39	104.00	104.00	104.00	22.830000	24.799999	104.00	66.3099
...	...	...	...	...	...	...	...	...	...	
217867	2001-04-01 00:00:00	10.45	1.81	104.00	104.00	104.00	73.000000	264.399994	104.00	5.2000
217868	2001-04-01 00:00:00	5.20	0.69	4.56	104.00	0.13	71.080002	129.300003	104.00	13.4600
217869	2001-04-01 00:00:00	0.49	1.09	104.00	1.00	0.19	76.279999	128.399994	0.35	5.0200
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.38	104.00	80.019997	197.000000	2.58	5.8400
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.04	0.18	76.809998	206.300003	5.20	8.3400

217872 rows × 16 columns

In [48]:

```
b.columns
```

Out[48]:

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

In [49]:

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

Out[49]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	2001-08-01 01:00:00	104.00	0.37	104.00	104.00	104.00	58.400002	87.150002	104.00	34.529999
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000
2	2001-08-01 01:00:00	104.00	0.28	104.00	104.00	104.00	50.660000	61.380001	104.00	46.310001
3	2001-08-01 01:00:00	104.00	0.47	104.00	104.00	104.00	69.790001	73.449997	104.00	40.650002
4	2001-08-01 01:00:00	104.00	0.39	104.00	104.00	104.00	22.830000	24.799999	104.00	66.309998
5	2001-08-01 01:00:00	2.11	0.63	2.48	5.94	0.05	66.260002	118.099998	3.15	33.500000
6	2001-08-01 01:00:00	104.00	0.28	104.00	104.00	104.00	35.799999	39.590000	104.00	68.250000
7	2001-08-01 01:00:00	104.00	0.67	104.00	104.00	104.00	74.830002	112.000000	104.00	26.410000
8	2001-08-01 01:00:00	104.00	0.41	104.00	104.00	104.00	33.209999	37.299999	104.00	62.299999
9	2001-08-01 01:00:00	104.00	0.17	104.00	104.00	0.13	24.129999	36.970001	104.00	46.200001

In [50]:

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

Out[50]:

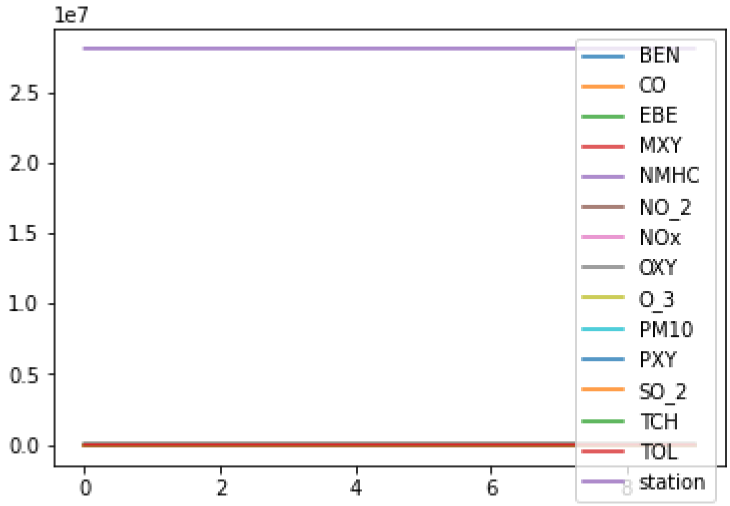
	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	104.00	0.37	104.00	104.00	104.00	58.400002	87.150002	104.00	34.529999	105.000000
1	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	100.599998
2	104.00	0.28	104.00	104.00	104.00	50.660000	61.380001	104.00	46.310001	100.099998
3	104.00	0.47	104.00	104.00	104.00	69.790001	73.449997	104.00	40.650002	69.779999
4	104.00	0.39	104.00	104.00	104.00	22.830000	24.799999	104.00	66.309998	75.180000
5	2.11	0.63	2.48	5.94	0.05	66.260002	118.099998	3.15	33.500000	122.699997
6	104.00	0.28	104.00	104.00	104.00	35.799999	39.590000	104.00	68.250000	124.900002
7	104.00	0.67	104.00	104.00	104.00	74.830002	112.000000	104.00	26.410000	113.000000
8	104.00	0.41	104.00	104.00	104.00	33.209999	37.299999	104.00	62.299999	125.300003
9	104.00	0.17	104.00	104.00	0.13	24.129999	36.970001	104.00	46.200001	95.589996

In [51]:

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

Out[51]:

```
<AxesSubplot:>
```

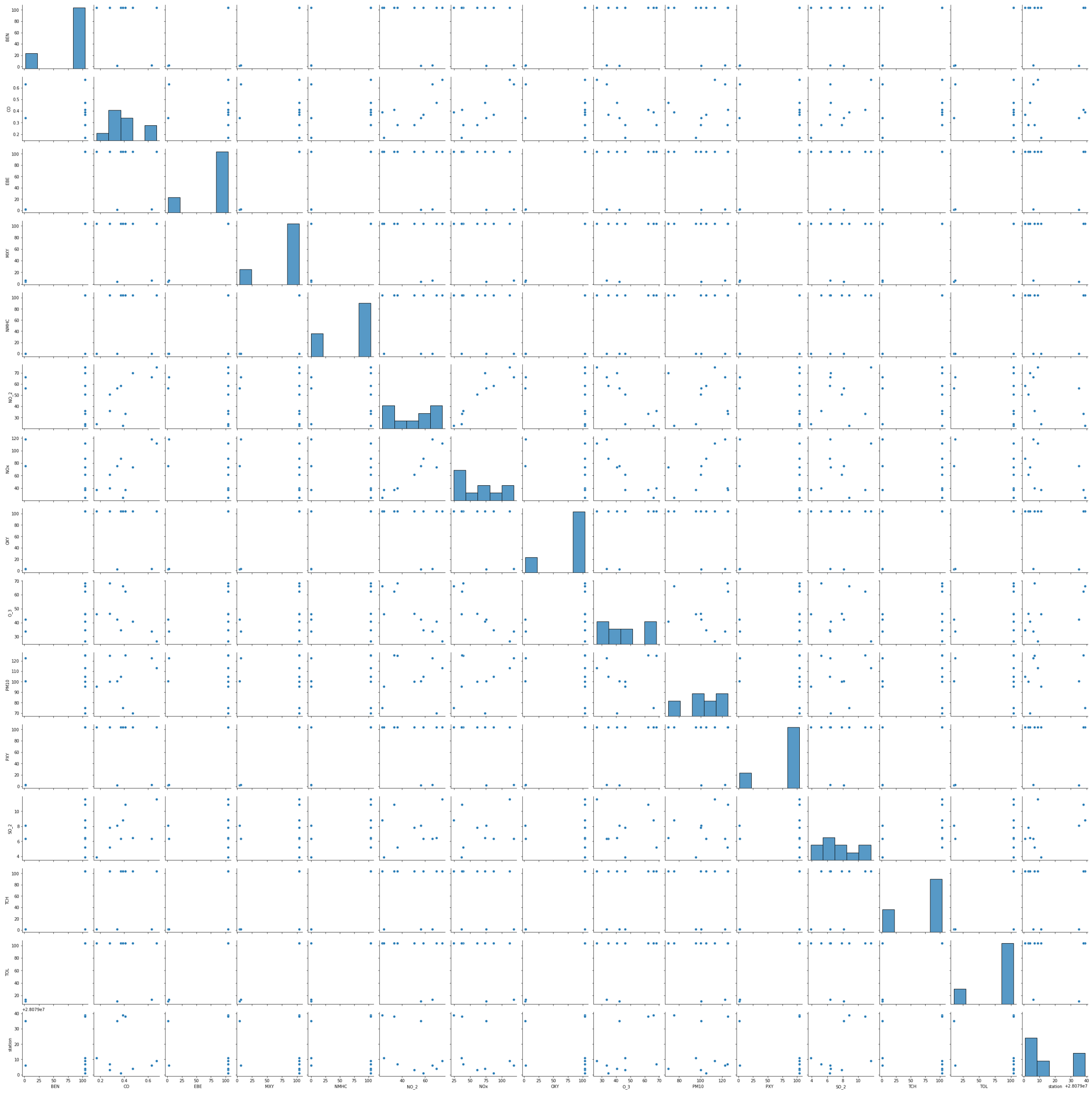


In [52]:

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

Out[52]:

```
<seaborn.axisgrid.PairGrid at 0x117b6a4c850>
```



In [53]:

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

In [54]:

```
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 [55]:

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

Out[55]:

```
LinearRegression()
```

In [56]:

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

```
1.1677154597820447
```

In [57]:

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

Out[57]:

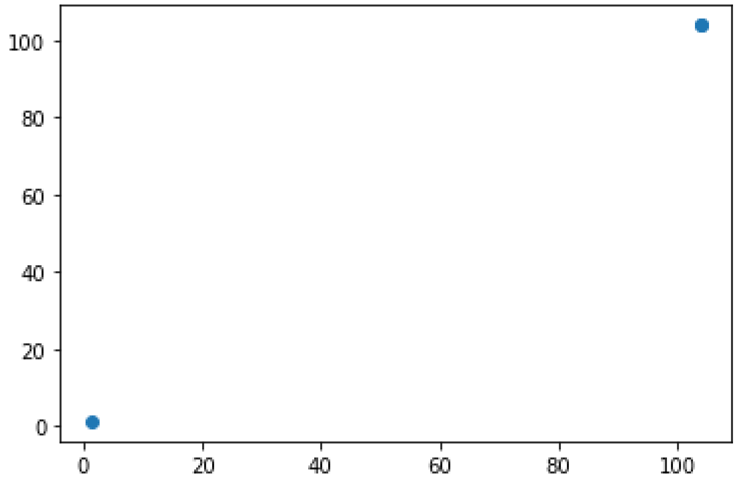
	Co-efficient
BEN	3.486930e-04
CO	-5.998829e-14
EBE	3.487270e-04
MXY	3.398481e-04
NMHC	9.873881e-01
NO_2	1.330907e-15
NOx	-2.523853e-16
OXY	3.466179e-04

In [58]:

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

Out[58]:

```
<matplotlib.collections.PathCollection at 0x117c498c880>
```



In [59]:

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

```
0.9999999904409993
```

In [60]:

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

In [61]:

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

Out[61]:

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

In [62]:

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

Out[62]:

```
0.9999346662259556
```

In [63]:

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

Out[63]:

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

In [64]:

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

Out[64]:

```
0.9999723842015672
```

In [65]:

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

Out[65]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O
0	2001-08-01 01:00:00	104.00	0.37	104.000000	104.0	104.00	58.400002	87.150002	104.00	34.5299
1	2001-08-01 01:00:00	1.50	0.34	1.490000	4.1	0.07	56.250000	75.169998	2.11	42.1600
2	2001-08-01 01:00:00	104.00	0.28	104.000000	104.0	104.00	50.660000	61.380001	104.00	46.3100
3	2001-08-01 01:00:00	104.00	0.47	104.000000	104.0	104.00	69.790001	73.449997	104.00	40.6500
4	2001-08-01 01:00:00	104.00	0.39	104.000000	104.0	104.00	22.830000	24.799999	104.00	66.3099
...	...	...	...	...	...	...	...	...	...	
6995	2001-08-13 04:00:00	104.00	0.00	104.000000	104.0	0.08	18.580000	18.590000	104.00	56.6600
6996	2001-08-13 04:00:00	104.00	0.09	104.000000	104.0	104.00	29.580000	32.770000	104.00	52.7099
6997	2001-08-13 04:00:00	1.38	0.17	30.530001	104.0	0.25	54.880001	68.870003	104.00	23.2400
6998	2001-08-13 04:00:00	104.00	0.01	104.000000	104.0	104.00	19.580000	20.990000	104.00	51.2700
6999	2001-08-13 04:00:00	104.00	0.00	104.000000	104.0	0.05	17.200001	18.219999	104.00	38.0900

7000 rows × 16 columns

In [66]:

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

In [67]:

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

In [68]:

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

In [69]:

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

Out[69]:

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

In [70]:

```
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 [71]:

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

In [72]:

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

```
[28079021]
```

In [73]:

```
logr.classes_
```

Out[73]:

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

In [74]:

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

Out[74]:

```
2.5317975595681163e-272
```

In [75]:

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

Out[75]:

```
5.035691156896021e-137
```

In [76]:

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

Out[76]:

```
0.6180952380952381
```

In [77]:

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

Out[77]:

```
ElasticNet()
```

In [78]:

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

```
[ 2.33990032e-06  0.00000000e+00  3.88046280e-08  1.33786963e-03
  9.86952627e-01  0.00000000e+00 -0.00000000e+00  4.41688852e-05]
```

In [79]:

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

```
1.199683583607822
```

In [80]:

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

```
0.9999998455533303
```

In [81]:

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

Out[81]:

```
RandomForestClassifier()
```

In [82]:

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

In [83]:

```
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[83]:

```
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 [84]:

```
grid_search.best_score_
```

Out[84]:

```
0.6108163265306122
```

In [85]:

```
rfc_best=grid_search.best_estimator_
```

In [86]:

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

Out[86]:

```
[Text(2466.36, 2491.5, 'X[12] <= -0.077\ngini = 0.958\nsamples = 3165\nva
lue = [182, 239, 218, 209, 216, 196, 200, 196, 197, 207\n204, 197, 211, 1
92, 196, 216, 220, 190, 196, 200\n198, 196, 205, 219]'),
 Text(1428.48, 2038.5, 'X[0] <= -0.349\ngini = 0.909\nsamples = 1450\nval
ue = [0, 0, 0, 209, 209, 0, 183, 196, 0, 207, 0, 197\n0, 0, 0, 216, 220,
0, 196, 0, 0, 0, 205, 219]'),
 Text(714.24, 1585.5, 'X[13] <= -1.729\ngini = 0.8\nsamples = 668\nvalue
= [0, 0, 0, 209, 0, 0, 0, 0, 0, 207, 0, 0, 0, 0\n0, 0, 220, 0, 196, 0, 0,
0, 0, 219]'),
 Text(357.12, 1132.5, 'X[12] <= -1.097\ngini = 0.27\nsamples = 121\nvalue
= [0, 0, 0, 2, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0\n0, 0, 158, 0, 23, 0, 0, 0,
0, 0]'),
 Text(178.56, 679.5, 'X[7] <= -1.984\ngini = 0.147\nsamples = 19\nvalue =
[0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 23, 0, 0, 0, 0,
0]'),
 Text(89.28, 226.5, 'gini = 0.0\nsamples = 14\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 19, 0, 0, 0, 0, 0]'),
 Text(267.84000000000003, 226.5, 'gini = 0.444\nsamples = 5\nvalue = [0,
```

In [ ]: