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 [748]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2018.csv")
a
```

Out[748]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_2
0	2018-03-01 01:00:00	NaN	NaN	0.3	NaN	NaN	1.0	29.0	31.0	NaN	NaN	NaN	2.0
1	2018-03-01 01:00:00	0.5	1.39	0.3	0.2	0.02	6.0	40.0	49.0	52.0	5.0	4.0	3.0
2	2018-03-01 01:00:00	0.4	NaN	NaN	0.2	NaN	4.0	41.0	47.0	NaN	NaN	NaN	NaN
3	2018-03-01 01:00:00	NaN	NaN	0.3	NaN	NaN	1.0	35.0	37.0	54.0	NaN	NaN	NaN
4	2018-03-01 01:00:00	NaN	NaN	NaN	NaN	NaN	1.0	27.0	29.0	49.0	NaN	NaN	3.0
...	...	...	...	...	...	...	...	...	...	...	...	...	..
69091	2018-02-01 00:00:00	NaN	NaN	0.5	NaN	NaN	66.0	91.0	192.0	1.0	35.0	22.0	NaN
69092	2018-02-01 00:00:00	NaN	NaN	0.7	NaN	NaN	87.0	107.0	241.0	NaN	29.0	NaN	15.0
69093	2018-02-01 00:00:00	NaN	NaN	NaN	NaN	NaN	28.0	48.0	91.0	2.0	NaN	NaN	NaN
69094	2018-02-01 00:00:00	NaN	NaN	NaN	NaN	NaN	141.0	103.0	320.0	2.0	NaN	NaN	NaN
69095	2018-02-01 00:00:00	NaN	NaN	NaN	NaN	NaN	69.0	96.0	202.0	3.0	26.0	NaN	NaN

69096 rows × 16 columns

In [749]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 69096 entries, 0 to 69095
Data columns (total 16 columns):
 #   Column   Non-Null Count  Dtype  
---  ------   --------------  -----  
 0   date     69096 non-null  object 
 1   BEN      16950 non-null  float64
 2   CH4      8440 non-null   float64
 3   CO       28598 non-null  float64
 4   EBE      16949 non-null  float64
 5   NMHC     8440 non-null   float64
 6   NO       68826 non-null  float64
 7   NO_2     68826 non-null  float64
 8   NOx      68826 non-null  float64
 9   O_3      40049 non-null  float64
 10  PM10     36911 non-null  float64
 11  PM25     18912 non-null  float64
 12  SO_2     28586 non-null  float64
 13  TCH      8440 non-null   float64
 14  TOL      16950 non-null  float64
 15  station  69096 non-null  int64  
dtypes: float64(14), int64(1), object(1)
memory usage: 8.4+ MB
```

In [750]:

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

Out[750]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25
0	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	29.0	31.0	104.0	104.0	104.0
1	2018-03-01 01:00:00	0.5	1.39	0.3	0.2	0.02	6.0	40.0	49.0	52.0	5.0	4.0
2	2018-03-01 01:00:00	0.4	104.00	104.0	0.2	104.00	4.0	41.0	47.0	104.0	104.0	104.0
3	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	35.0	37.0	54.0	104.0	104.0
4	2018-03-01 01:00:00	104.0	104.00	104.0	104.0	104.00	1.0	27.0	29.0	49.0	104.0	104.0
...	...	...	...	...	...	...	...	...	...	...	...	...
69091	2018-02-01 00:00:00	104.0	104.00	0.5	104.0	104.00	66.0	91.0	192.0	1.0	35.0	22.0
69092	2018-02-01 00:00:00	104.0	104.00	0.7	104.0	104.00	87.0	107.0	241.0	104.0	29.0	104.0
69093	2018-02-01 00:00:00	104.0	104.00	104.0	104.0	104.00	28.0	48.0	91.0	2.0	104.0	104.0
69094	2018-02-01 00:00:00	104.0	104.00	104.0	104.0	104.00	141.0	103.0	320.0	2.0	104.0	104.0
69095	2018-02-01 00:00:00	104.0	104.00	104.0	104.0	104.00	69.0	96.0	202.0	3.0	26.0	104.0

69096 rows × 16 columns

In [751]:

```
b.columns
```

Out[751]:

```
Index(['date', 'BEN', 'CH4', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'NOx', 'O_
3',
       'PM10', 'PM25', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [752]:

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

Out[752]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_2
0	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	29.0	31.0	104.0	104.0	104.0	2.0
1	2018-03-01 01:00:00	0.5	1.39	0.3	0.2	0.02	6.0	40.0	49.0	52.0	5.0	4.0	3.0
2	2018-03-01 01:00:00	0.4	104.00	104.0	0.2	104.00	4.0	41.0	47.0	104.0	104.0	104.0	104.0
3	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	35.0	37.0	54.0	104.0	104.0	104.0
4	2018-03-01 01:00:00	104.0	104.00	104.0	104.0	104.00	1.0	27.0	29.0	49.0	104.0	104.0	3.0
5	2018-03-01 01:00:00	0.3	104.00	0.3	0.2	104.00	1.0	27.0	29.0	57.0	8.0	104.0	6.0
6	2018-03-01 01:00:00	0.4	1.11	0.2	0.1	0.06	1.0	25.0	27.0	55.0	5.0	4.0	4.0
7	2018-03-01 01:00:00	104.0	104.00	104.0	104.0	104.00	1.0	37.0	39.0	54.0	104.0	104.0	104.0
8	2018-03-01 01:00:00	104.0	104.00	0.5	104.0	104.00	3.0	43.0	47.0	29.0	104.0	104.0	5.0
9	2018-03-01 01:00:00	104.0	104.00	0.2	104.0	104.00	2.0	26.0	29.0	104.0	4.0	104.0	6.0

In [753]:

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

Out[753]:

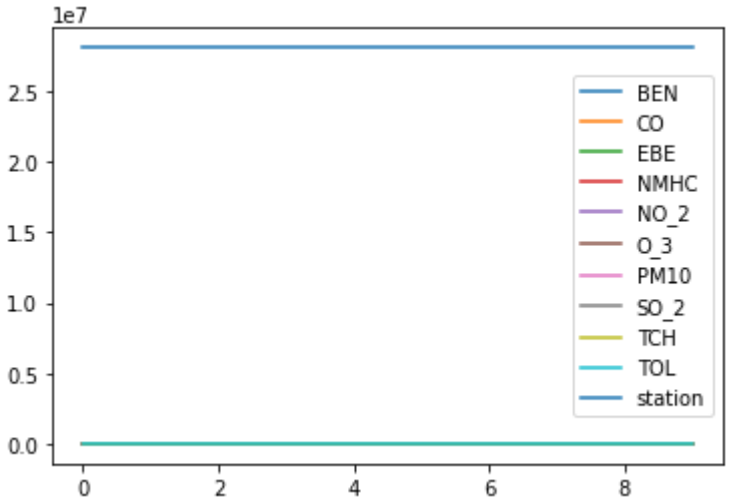
	BEN	CO	EBE	NMHC	NO_2	O_3	PM10	SO_2	TCH	TOL	station
0	104.0	0.3	104.0	104.00	29.0	104.0	104.0	2.0	104.00	104.0	28079004
1	0.5	0.3	0.2	0.02	40.0	52.0	5.0	3.0	1.41	0.8	28079008
2	0.4	104.0	0.2	104.00	41.0	104.0	104.0	104.0	104.00	1.1	28079011
3	104.0	0.3	104.0	104.00	35.0	54.0	104.0	104.0	104.00	104.0	28079016
4	104.0	104.0	104.0	104.00	27.0	49.0	104.0	3.0	104.00	104.0	28079017
5	0.3	0.3	0.2	104.00	27.0	57.0	8.0	6.0	104.00	1.0	28079018
6	0.4	0.2	0.1	0.06	25.0	55.0	5.0	4.0	1.16	1.4	28079024
7	104.0	104.0	104.0	104.00	37.0	54.0	104.0	104.0	104.00	104.0	28079027
8	104.0	0.5	104.0	104.00	43.0	29.0	104.0	5.0	104.00	104.0	28079035
9	104.0	0.2	104.0	104.00	26.0	104.0	4.0	6.0	104.00	104.0	28079036

In [754]:

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

Out[754]:

```
<AxesSubplot:>
```

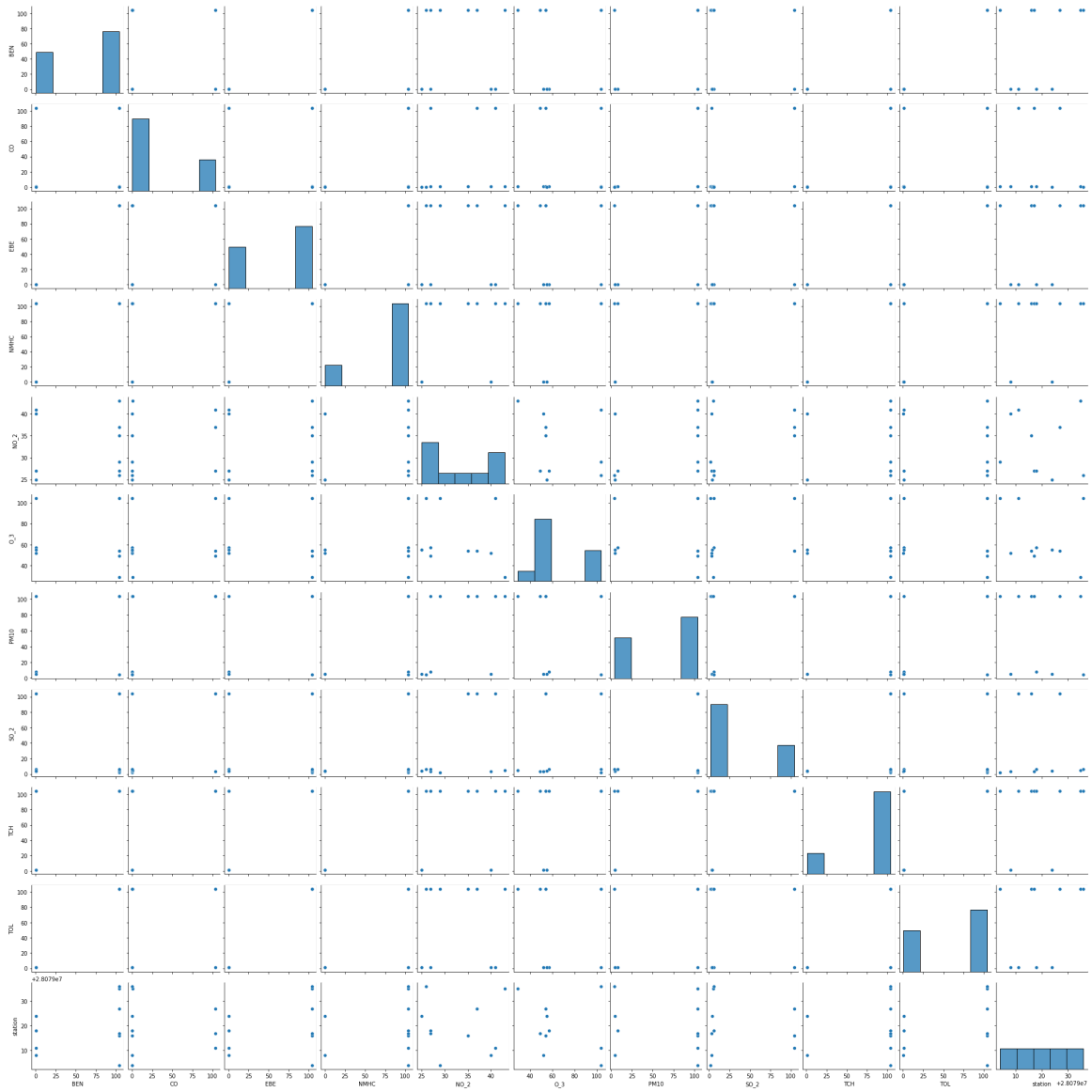


In [755]:

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

Out[755]:

```
<seaborn.axisgrid.PairGrid at 0x118af61a5b0>
```



In [756]:

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

In [757]:

```
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 [758]:

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

Out[758]:

```
LinearRegression()
```

In [759]:

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

```
1.1006349496499297
```

In [760]:

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

Out[760]:

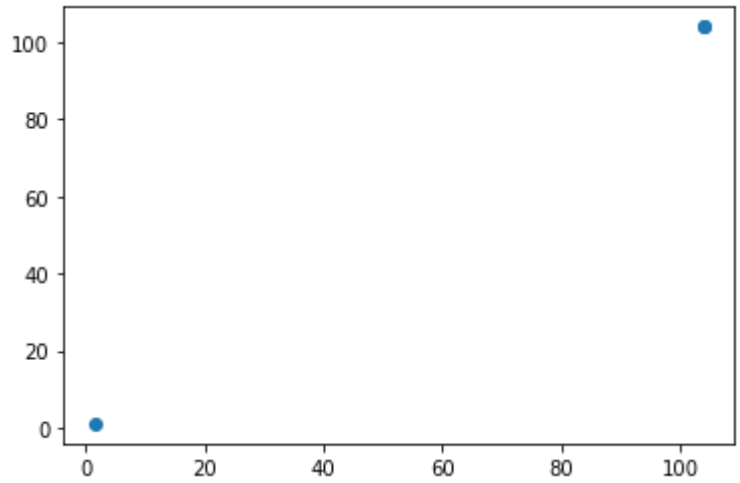
	Co-efficient
BEN	1.005093e-13
CO	-8.217196e-17
EBE	-1.003976e-13
NMHC	9.894170e-01
NO_2	-1.647502e-15

In [761]:

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

Out[761]:

```
<matplotlib.collections.PathCollection at 0x118b8a698e0>
```



In [762]:

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

```
0.9999880488867369
```

In [763]:

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

In [764]:

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

Out[764]:

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

In [765]:

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

Out[765]:

```
0.9999974127809635
```

In [766]:

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

Out[766]:

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

In [767]:

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

Out[767]:

```
0.9999753721840651
```

In [768]:

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

Out[768]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO
0	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	29.0	31.0	104.0	104.0	104.0	2
1	2018-03-01 01:00:00	0.5	1.39	0.3	0.2	0.02	6.0	40.0	49.0	52.0	5.0	4.0	3
2	2018-03-01 01:00:00	0.4	104.00	104.0	0.2	104.00	4.0	41.0	47.0	104.0	104.0	104.0	104
3	2018-03-01 01:00:00	104.0	104.00	0.3	104.0	104.00	1.0	35.0	37.0	54.0	104.0	104.0	104
4	2018-03-01 01:00:00	104.0	104.00	104.0	104.0	104.00	1.0	27.0	29.0	49.0	104.0	104.0	3
...	...	...	...	...	...	...	...	...	...	...	...	...	
6995	2018-03-13 04:00:00	104.0	104.00	0.2	104.0	104.00	1.0	9.0	11.0	60.0	104.0	104.0	104
6996	2018-03-13 04:00:00	104.0	104.00	104.0	104.0	104.00	1.0	38.0	39.0	104.0	15.0	104.0	3
6997	2018-03-13 04:00:00	104.0	104.00	104.0	104.0	104.00	1.0	17.0	18.0	104.0	8.0	3.0	104
6998	2018-03-13 04:00:00	104.0	104.00	104.0	104.0	104.00	1.0	14.0	16.0	104.0	7.0	5.0	104
6999	2018-03-13 04:00:00	104.0	104.00	104.0	104.0	104.00	1.0	10.0	11.0	49.0	104.0	104.0	104

7000 rows × 16 columns

In [769]:

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

In [770]:

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

In [771]:

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

In [772]:

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

Out[772]:

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

In [773]:

```
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 [774]:

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

In [775]:

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

```
[28079050]
```

In [776]:

```
logr.classes_
```

Out[776]:

```
array([28079004, 28079008, 28079011, 28079016, 28079017, 28079018,
       28079024, 28079027, 28079035, 28079036, 28079038, 28079039,
       28079040, 28079047, 28079048, 28079049, 28079050, 28079054,
       28079055, 28079056, 28079057, 28079058, 28079059, 28079060],
      dtype=int64)
```

In [777]:

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

Out[777]:

```
0.0
```

In [778]:

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

Out[778]:

```
0.0
```

In [779]:

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

Out[779]:

```
0.95
```

In [780]:

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

Out[780]:

```
ElasticNet()
```

In [781]:

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

```
[7.90496910e-05 0.00000000e+00 8.33668684e-07 9.88612279e-01
 0.00000000e+00]
```

In [782]:

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

```
1.167614612208311
```

In [783]:

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

```
0.9999929195096
```

In [784]:

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

Out[784]:

```
RandomForestClassifier()
```

In [785]:

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

In [786]:

```
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[786]:

```
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 [787]:

```
grid_search.best_score_
```

Out[787]:

```
0.9955102040816326
```

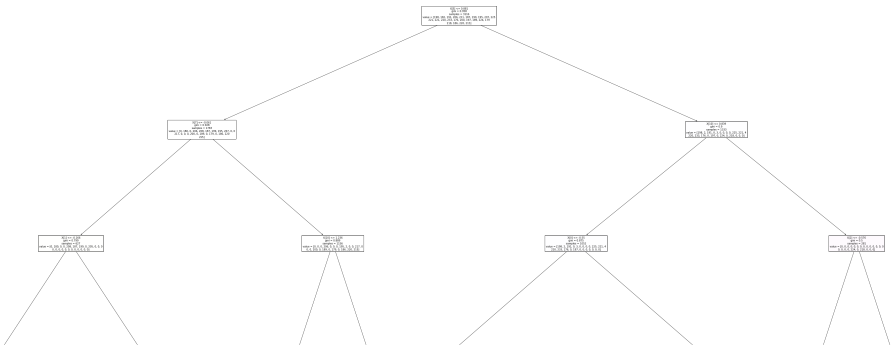
In [788]:

```
rfc_best=grid_search.best_estimator_
```

In [789]:

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

```
0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 98, 0, 0, 0]'),
 Text(4166.400000000001, 679.5, 'X[6] <= -0.923\ngini = 0.048\nsamples =
79\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 3, 0,
120, 0, 0, 0]'),
 Text(4017.6000000000004, 226.5, 'gini = 0.0\nsamples = 31\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 57, 0, 0, 0]'),
 Text(4315.200000000001, 226.5, 'gini = 0.087\nsamples = 48\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 3, 0, 63, 0, 0, 0]')]
```



In [ ]: