In [652]:
```
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 [653]:
```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\csvs_per_year\csvs_per_year\madrid_2014
a
```

Out[653]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2014-06-01 01:00:00	NaN	0.2	NaN	NaN	3.0	10.0	NaN	NaN	NaN	3.0	NaN	NaN	28079004
1	2014-06-01 01:00:00	0.2	0.2	0.1	0.11	3.0	17.0	68.0	10.0	5.0	5.0	1.36	1.3	28079008
2	2014-06-01 01:00:00	0.3	NaN	0.1	NaN	2.0	6.0	NaN	NaN	NaN	NaN	NaN	1.1	28079011
3	2014-06-01 01:00:00	NaN	0.2	NaN	NaN	1.0	6.0	79.0	NaN	NaN	NaN	NaN	NaN	28079016
4	2014-06-01 01:00:00	NaN	NaN	NaN	NaN	1.0	6.0	75.0	NaN	NaN	4.0	NaN	NaN	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210019	2014-09-01 00:00:00	NaN	0.5	NaN	NaN	20.0	84.0	29.0	NaN	NaN	NaN	NaN	NaN	28079056
210020	2014-09-01 00:00:00	NaN	0.3	NaN	NaN	1.0	22.0	NaN	15.0	NaN	6.0	NaN	NaN	28079057
210021	2014-09-01 00:00:00	NaN	NaN	NaN	NaN	1.0	13.0	70.0	NaN	NaN	NaN	NaN	NaN	28079058
210022	2014-09-01 00:00:00	NaN	NaN	NaN	NaN	3.0	38.0	42.0	NaN	NaN	NaN	NaN	NaN	28079059
210023	2014-09-01 00:00:00	NaN	NaN	NaN	NaN	1.0	26.0	65.0	11.0	NaN	NaN	NaN	NaN	28079060

210024 rows × 14 columns

In [654]:
```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 210024 entries, 0 to 210023
Data columns (total 14 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     210024 non-null  object 
 1   BEN      46703 non-null   float64
 2   CO       87023 non-null   float64
 3   EBE      46722 non-null   float64
 4   NMHC     25021 non-null   float64
 5   NO       209154 non-null  float64
 6   NO_2     209154 non-null  float64
 7   O_3      121681 non-null  float64
 8   PM10     104311 non-null  float64
 9   PM25     51954 non-null   float64
 10  SO_2     87141 non-null   float64
 11  TCH      25021 non-null   float64
 12  TOL      46570 non-null   float64
 13  station  210024 non-null  int64  
dtypes: float64(12), int64(1), object(1)
memory usage: 22.4+ MB
```

In [655]:
```
b=a.fillna(value=86)
b
```

Out[655]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	3.0	10.0	86.0	86.0	86.0	3.0	86.00	86.0	28079004
1	2014-06-01 01:00:00	0.2	0.2	0.1	0.11	3.0	17.0	68.0	10.0	5.0	5.0	1.36	1.3	28079008
2	2014-06-01 01:00:00	0.3	86.0	0.1	86.00	2.0	6.0	86.0	86.0	86.0	86.0	86.00	1.1	28079011
3	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	6.0	79.0	86.0	86.0	86.0	86.00	86.0	28079016
4	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	6.0	75.0	86.0	86.0	4.0	86.00	86.0	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210019	2014-09-01 00:00:00	86.0	0.5	86.0	86.00	20.0	84.0	29.0	86.0	86.0	86.0	86.00	86.0	28079056
210020	2014-09-01 00:00:00	86.0	0.3	86.0	86.00	1.0	22.0	86.0	15.0	86.0	6.0	86.00	86.0	28079057
210021	2014-09-01 00:00:00	86.0	86.0	86.0	86.00	1.0	13.0	70.0	86.0	86.0	86.0	86.00	86.0	28079058
210022	2014-09-01 00:00:00	86.0	86.0	86.0	86.00	3.0	38.0	42.0	86.0	86.0	86.0	86.00	86.0	28079059
210023	2014-09-01 00:00:00	86.0	86.0	86.0	86.00	1.0	26.0	65.0	11.0	86.0	86.0	86.00	86.0	28079060

210024 rows × 14 columns

In [656]:
```
b.columns
```

Out[656]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'O_3', 'PM10', 'PM25',
       'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [657]:
```
c=b.head(30)
c
```

Out[657]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	3.0	10.0	86.0	86.0	86.0	3.0	86.00	86.0	28079004
1	2014-06-01 01:00:00	0.2	0.2	0.1	0.11	3.0	17.0	68.0	10.0	5.0	5.0	1.36	1.3	28079008
2	2014-06-01 01:00:00	0.3	86.0	0.1	86.00	2.0	6.0	86.0	86.0	86.0	86.0	86.00	1.1	28079011
3	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	6.0	79.0	86.0	86.0	86.0	86.00	86.0	28079016
4	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	6.0	75.0	86.0	86.0	4.0	86.00	86.0	28079017
5	2014-06-01 01:00:00	0.1	0.4	0.1	86.00	1.0	10.0	83.0	7.0	86.0	2.0	86.00	0.2	28079018
6	2014-06-01 01:00:00	0.1	0.2	0.1	0.23	1.0	5.0	80.0	4.0	3.0	2.0	1.21	0.1	28079024
7	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	1.0	86.0	86.0	86.0	86.0	86.00	86.0	28079027
8	2014-06-01 01:00:00	86.0	0.3	86.0	86.00	5.0	22.0	68.0	86.0	86.0	4.0	86.00	86.0	28079035
9	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	4.0	86.0	14.0	86.0	1.0	86.00	86.0	28079036
10	2014-06-01 01:00:00	0.1	86.0	0.1	86.00	6.0	18.0	86.0	8.0	5.0	2.0	86.00	0.7	28079038
11	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	7.0	81.0	86.0	86.0	86.0	86.00	86.0	28079039
12	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	3.0	13.0	86.0	3.0	86.0	4.0	86.00	86.0	28079040
13	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	3.0	10.0	86.0	11.0	6.0	86.0	86.00	86.0	28079047
14	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	8.0	86.0	5.0	1.0	86.0	86.00	86.0	28079048
15	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	7.0	75.0	86.0	86.0	86.0	86.00	86.0	28079049
16	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	7.0	86.0	7.0	6.0	86.0	86.00	86.0	28079050
17	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	10.0	78.0	86.0	86.0	86.0	86.00	86.0	28079054
18	2014-06-01 01:00:00	86.0	86.0	86.0	0.23	1.0	4.0	86.0	8.0	86.0	86.0	1.20	86.0	28079055
19	2014-06-01 01:00:00	86.0	0.3	86.0	86.00	7.0	28.0	65.0	86.0	86.0	86.0	86.00	86.0	28079056
20	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	4.0	86.0	8.0	86.0	6.0	86.00	86.0	28079057
21	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	4.0	64.0	86.0	86.0	86.0	86.00	86.0	28079058
22	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	2.0	3.0	69.0	86.0	86.0	86.0	86.00	86.0	28079059
23	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	5.0	86.0	7.0	86.0	86.0	86.00	86.0	28079060

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
24	2014-06-01 02:00:00	86.0	0.1	86.0	86.00	2.0	9.0	86.0	86.0	86.0	3.0	86.00	86.0	28079004
25	2014-06-01 02:00:00	0.2	0.2	0.1	0.11	4.0	21.0	63.0	9.0	6.0	5.0	1.36	0.8	28079008
26	2014-06-01 02:00:00	0.3	86.0	0.1	86.00	1.0	2.0	86.0	86.0	86.0	86.0	86.00	0.9	28079011
27	2014-06-01 02:00:00	86.0	0.2	86.0	86.00	1.0	6.0	79.0	86.0	86.0	86.0	86.00	86.0	28079016
28	2014-06-01 02:00:00	86.0	86.0	86.0	86.00	1.0	4.0	76.0	86.0	86.0	5.0	86.00	86.0	28079017
29	2014-06-01 02:00:00	0.1	0.4	0.1	86.00	1.0	13.0	74.0	8.0	86.0	2.0	86.00	0.1	28079018

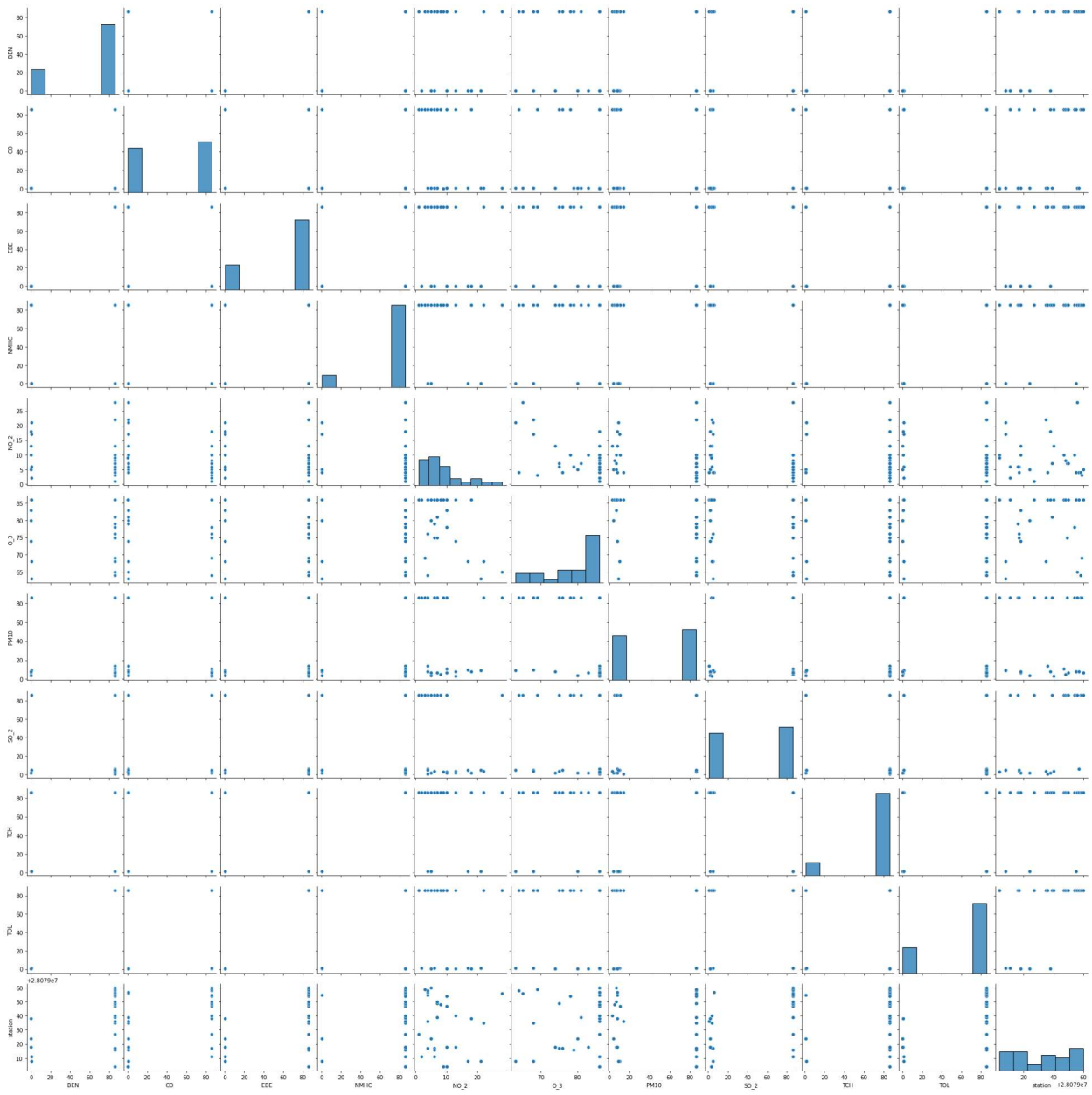
In [658]:
```
d=c[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
d
```

Out[658]:

	BEN	CO	EBE	NMHC	NO_2	O_3	PM10	SO_2	TCH	TOL	station
0	86.0	0.2	86.0	86.00	10.0	86.0	86.0	3.0	86.00	86.0	28079004
1	0.2	0.2	0.1	0.11	17.0	68.0	10.0	5.0	1.36	1.3	28079008
2	0.3	86.0	0.1	86.00	6.0	86.0	86.0	86.0	86.00	1.1	28079011
3	86.0	0.2	86.0	86.00	6.0	79.0	86.0	86.0	86.00	86.0	28079016
4	86.0	86.0	86.0	86.00	6.0	75.0	86.0	4.0	86.00	86.0	28079017
5	0.1	0.4	0.1	86.00	10.0	83.0	7.0	2.0	86.00	0.2	28079018
6	0.1	0.2	0.1	0.23	5.0	80.0	4.0	2.0	1.21	0.1	28079024
7	86.0	86.0	86.0	86.00	1.0	86.0	86.0	86.0	86.00	86.0	28079027
8	86.0	0.3	86.0	86.00	22.0	68.0	86.0	4.0	86.00	86.0	28079035
9	86.0	0.2	86.0	86.00	4.0	86.0	14.0	1.0	86.00	86.0	28079036
10	0.1	86.0	0.1	86.00	18.0	86.0	8.0	2.0	86.00	0.7	28079038
11	86.0	0.2	86.0	86.00	7.0	81.0	86.0	86.0	86.00	86.0	28079039
12	86.0	86.0	86.0	86.00	13.0	86.0	3.0	4.0	86.00	86.0	28079040
13	86.0	86.0	86.0	86.00	10.0	86.0	11.0	86.0	86.00	86.0	28079047
14	86.0	86.0	86.0	86.00	8.0	86.0	5.0	86.0	86.00	86.0	28079048
15	86.0	86.0	86.0	86.00	7.0	75.0	86.0	86.0	86.00	86.0	28079049
16	86.0	86.0	86.0	86.00	7.0	86.0	7.0	86.0	86.00	86.0	28079050
17	86.0	86.0	86.0	86.00	10.0	78.0	86.0	86.0	86.00	86.0	28079054
18	86.0	86.0	86.0	0.23	4.0	86.0	8.0	86.0	1.20	86.0	28079055
19	86.0	0.3	86.0	86.00	28.0	65.0	86.0	86.0	86.00	86.0	28079056
20	86.0	0.2	86.0	86.00	4.0	86.0	8.0	6.0	86.00	86.0	28079057
21	86.0	86.0	86.0	86.00	4.0	64.0	86.0	86.0	86.00	86.0	28079058
22	86.0	86.0	86.0	86.00	3.0	69.0	86.0	86.0	86.00	86.0	28079059
23	86.0	86.0	86.0	86.00	5.0	86.0	7.0	86.0	86.00	86.0	28079060
24	86.0	0.1	86.0	86.00	9.0	86.0	86.0	3.0	86.00	86.0	28079004
25	0.2	0.2	0.1	0.11	21.0	63.0	9.0	5.0	1.36	0.8	28079008
26	0.3	86.0	0.1	86.00	2.0	86.0	86.0	86.0	86.00	0.9	28079011
27	86.0	0.2	86.0	86.00	6.0	79.0	86.0	86.0	86.00	86.0	28079016
28	86.0	86.0	86.0	86.00	4.0	76.0	86.0	5.0	86.00	86.0	28079017
29	0.1	0.4	0.1	86.00	13.0	74.0	8.0	2.0	86.00	0.1	28079018

In [659]:
```
sns.pairplot(d)
```

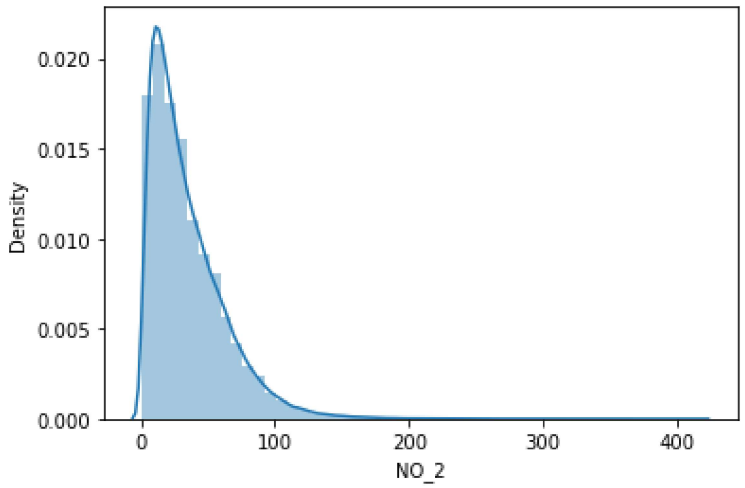
Out[659]: <seaborn.axisgrid.PairGrid at 0x1b71ec9e130>



In [660]:
```
sns.distplot(a['NO_2'])
```

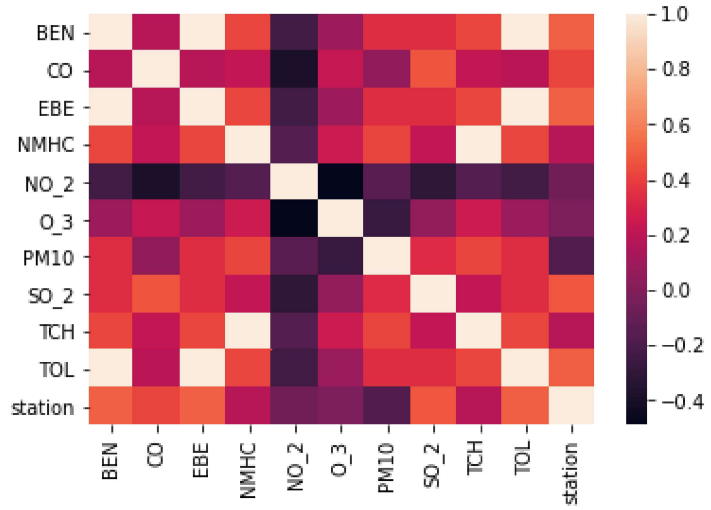
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[660]: <AxesSubplot:xlabel='NO_2', ylabel='Density'>



In [661]:
```
sns.heatmap(d.corr())
```

Out[661]: <AxesSubplot:>



In [662]:
```
x=d[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2']]
y=d['TCH']
```

In [663]:
```
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 [664]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[664]: LinearRegression()

In [665]:
```
print(lr.intercept_)
```

```
1.1110435929424654
```

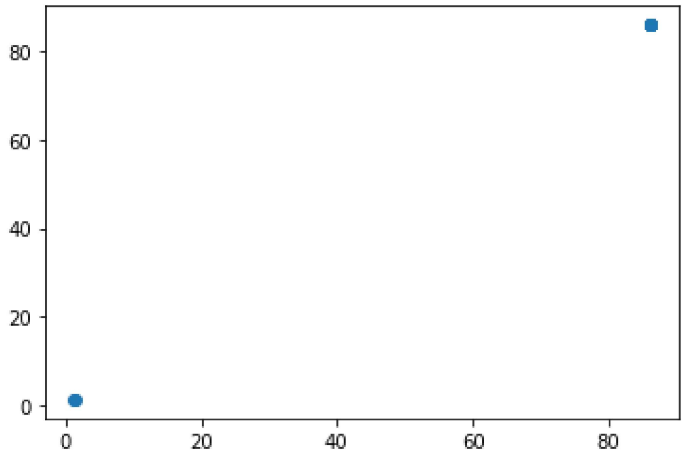
In [666]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[666]:

	Co-efficient
BEN	0.142538
CO	-0.000223
EBE	-0.142758
NMHC	0.987220
NO_2	0.001252

In [667]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[667]:
```
<matplotlib.collections.PathCollection at 0x1b72afc28b0>
```



In [668]:
```
print(lr.score(x_test,y_test))
```

```
0.999997386323644
```

In [669]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [670]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[670]:
```
Ridge(alpha=10)
```

In [671]:
```
rr.score(x_test,y_test)
```

Out[671]:
```
0.999995914178287
```

In [672]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[672]:
```
Lasso(alpha=10)
```

In [673]:
```
la.score(x_test,y_test)
```

Out[673]:
```
0.9997204464884435
```

In [674]:
```
a1=b.head(7000)
a1
```

Out[674]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	3.0	10.0	86.0	86.0	86.0	3.0	86.00	86.0	28079004
1	2014-06-01 01:00:00	0.2	0.2	0.1	0.11	3.0	17.0	68.0	10.0	5.0	5.0	1.36	1.3	28079008
2	2014-06-01 01:00:00	0.3	86.0	0.1	86.00	2.0	6.0	86.0	86.0	86.0	86.0	86.00	1.1	28079011
3	2014-06-01 01:00:00	86.0	0.2	86.0	86.00	1.0	6.0	79.0	86.0	86.0	86.0	86.00	86.0	28079016
4	2014-06-01 01:00:00	86.0	86.0	86.0	86.00	1.0	6.0	75.0	86.0	86.0	4.0	86.00	86.0	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6995	2014-06-13 04:00:00	86.0	0.2	86.0	86.00	1.0	16.0	63.0	86.0	86.0	86.0	86.00	86.0	28079039
6996	2014-06-13 04:00:00	86.0	86.0	86.0	86.00	3.0	18.0	86.0	22.0	86.0	4.0	86.00	86.0	28079040
6997	2014-06-13 04:00:00	86.0	86.0	86.0	86.00	2.0	17.0	86.0	22.0	15.0	86.0	86.00	86.0	28079047
6998	2014-06-13 04:00:00	86.0	86.0	86.0	86.00	1.0	14.0	86.0	29.0	14.0	86.0	86.00	86.0	28079048
6999	2014-06-13 04:00:00	86.0	86.0	86.0	86.00	3.0	14.0	59.0	86.0	86.0	86.0	86.00	86.0	28079049

7000 rows × 14 columns

In [675]:
```
e=a1[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
```

In [676]:
```
f=e.iloc[:,0:14]
g=e.iloc[:,-1]
```

In [677]:
```
h=StandardScaler().fit_transform(f)
```

In [678]:
```
logr=LogisticRegression(max_iter=10000)
logr.fit(h,g)
```

Out[678]:
```
LogisticRegression(max_iter=10000)
```

In [679]:
```
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 [680]:
```
i=[[10,20,30,40,50,60,15,26,37,47,58]]
```

In [681]:
```
prediction=logr.predict(i)
print(prediction)
```

```
[28079060]
```

In [682]:
```
logr.classes_
```

Out[682]:
```
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 [683]:
```
logr.predict_proba(i)[0][0]
```

Out[683]:
```
0.0
```

In [684]:
```
logr.predict_proba(i)[0][1]
```

Out[684]:
```
0.0
```

In [685]:
```
logr.score(h_test,g_test)
```

Out[685]:
```
0.9328571428571428
```

In [686]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[686]:
```
ElasticNet()
```

In [687]:
```
print(en.coef_)
```

```
[-0.00000000e+00 -2.29937927e-05 -0.00000000e+00  9.85502898e-01
  0.00000000e+00]
```

In [688]:
```
print(en.intercept_)
```

```
1.235000236806897
```

In [689]:
```
prediction=en.predict(x_test)
print(en.score(x_test,y_test))
```

```
0.9999941929868205
```

In [690]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(h_train,g_train)
```

Out[690]:
```
RandomForestClassifier()
```

In [691]:
```
parameters={'max_depth':[1,2,3,4,5],
 'min_samples_leaf':[5,10,15,20,25],
 'n_estimators':[10,20,30,40,50]
 }
```

In [692]:
```
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)
```

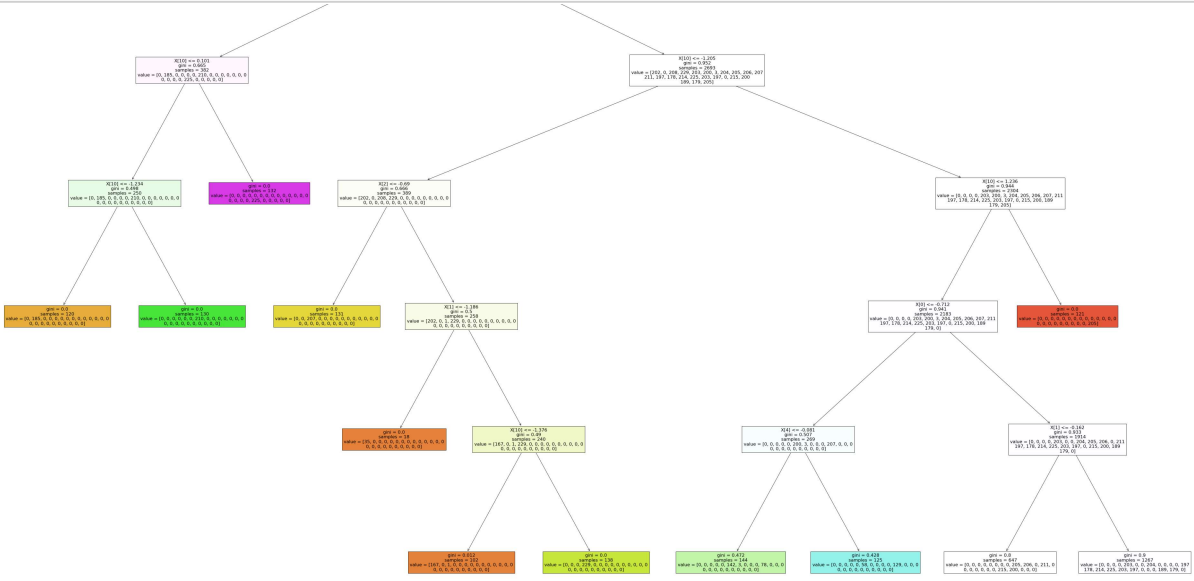
Out[692]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [693]:
```
grid_search.best_score_
```

Out[693]:
```
0.9953061224489796
```

In [694]:
```
rfc_best=grid_search.best_estimator_
```

In [695]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,50))
plot_tree(rfc_best.estimators_[2],filled=True)
```



Conclusion: from this data set i observed that the ridge has the highest accuracy of 0.9961224489795919

In [ ]: