In [1]:
```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
```

In [2]:
```python
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\csvs_per_year\csvs_per_year\ma
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2010-03-01 01:00:00	NaN	0.29	NaN	NaN	NaN	25.090000	29.219999	NaN	68.930000	
1	2010-03-01 01:00:00	NaN	0.27	NaN	NaN	NaN	24.879999	30.040001	NaN	NaN	
2	2010-03-01 01:00:00	NaN	0.28	NaN	NaN	NaN	17.410000	20.540001	NaN	72.120003	
3	2010-03-01 01:00:00	0.38	0.24	1.74	NaN	0.05	15.610000	21.080000	NaN	72.970001	19.
4	2010-03-01 01:00:00	0.79	NaN	1.32	NaN	NaN	21.430000	26.070000	NaN	NaN	24.
...	...	...	...	...	...	...	...	...	...	...	
209443	2010-08-01 00:00:00	NaN	0.55	NaN	NaN	NaN	125.000000	219.899994	NaN	25.379999	
209444	2010-08-01 00:00:00	NaN	0.27	NaN	NaN	NaN	45.709999	47.410000	NaN	NaN	51.
209445	2010-08-01 00:00:00	NaN	NaN	NaN	NaN	0.24	46.560001	49.040001	NaN	46.250000	
209446	2010-08-01 00:00:00	NaN	NaN	NaN	NaN	NaN	46.770000	50.119999	NaN	77.709999	
209447	2010-08-01 00:00:00	0.92	0.43	0.71	NaN	0.25	76.330002	88.190002	NaN	52.259998	47.

209448 rows × 17 columns

In [3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209448 entries, 0 to 209447
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     209448 non-null  object 
 1   BEN      60268 non-null   float64
 2   CO       94982 non-null   float64
 3   EBE      60253 non-null   float64
 4   MXY      6750 non-null    float64
 5   NMHC     51727 non-null   float64
 6   NO_2     208219 non-null  float64
 7   NOx      208210 non-null  float64
 8   OXY      6750 non-null    float64
 9   O_3      126684 non-null  float64
 10  PM10     106186 non-null  float64
 11  PM25     55514 non-null   float64
 12  PXY      6740 non-null    float64
 13  SO_2     93184 non-null   float64
 14  TCH      51730 non-null   float64
 15  TOL      60171 non-null   float64
 16  station  209448 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 27.2+ MB
```

In [4]:
```
b=a.fillna(value=111)
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O
0	2010-03-01 01:00:00	111.00	0.29	111.00	111.0	111.00	25.090000	29.219999	111.0	68.9300
1	2010-03-01 01:00:00	111.00	0.27	111.00	111.0	111.00	24.879999	30.040001	111.0	111.0000
2	2010-03-01 01:00:00	111.00	0.28	111.00	111.0	111.00	17.410000	20.540001	111.0	72.1200
3	2010-03-01 01:00:00	0.38	0.24	1.74	111.0	0.05	15.610000	21.080000	111.0	72.9700
4	2010-03-01 01:00:00	0.79	111.00	1.32	111.0	111.00	21.430000	26.070000	111.0	111.0000
...	...	...	...	...	...	...	...	...	...	
209443	2010-08-01 00:00:00	111.00	0.55	111.00	111.0	111.00	125.000000	219.899994	111.0	25.3799
209444	2010-08-01 00:00:00	111.00	0.27	111.00	111.0	111.00	45.709999	47.410000	111.0	111.0000
209445	2010-08-01 00:00:00	111.00	111.00	111.00	111.0	0.24	46.560001	49.040001	111.0	46.2500
209446	2010-08-01 00:00:00	111.00	111.00	111.00	111.0	111.00	46.770000	50.119999	111.0	77.7099
209447	2010-08-01 00:00:00	0.92	0.43	0.71	111.0	0.25	76.330002	88.190002	111.0	52.2599

209448 rows × 17 columns

In [5]:
```

```

Out[5]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3
',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [6]:
```
c=b.head(11)
```

Out[6]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2010-03-01 01:00:00	111.00	0.29	111.00	111.0	111.00	25.090000	29.219999	111.0	68.930000	111
1	2010-03-01 01:00:00	111.00	0.27	111.00	111.0	111.00	24.879999	30.040001	111.0	111.000000	111
2	2010-03-01 01:00:00	111.00	0.28	111.00	111.0	111.00	17.410000	20.540001	111.0	72.120003	111
3	2010-03-01 01:00:00	0.38	0.24	1.74	111.0	0.05	15.610000	21.080000	111.0	72.970001	19
4	2010-03-01 01:00:00	0.79	111.00	1.32	111.0	111.00	21.430000	26.070000	111.0	111.000000	24
5	2010-03-01 01:00:00	0.56	111.00	0.58	111.0	111.00	21.370001	25.870001	111.0	111.000000	111
6	2010-03-01 01:00:00	111.00	111.00	111.00	111.0	111.00	16.660000	25.230000	111.0	111.000000	39
7	2010-03-01 01:00:00	111.00	0.23	111.00	111.0	111.00	17.799999	21.639999	111.0	55.880001	111
8	2010-03-01 01:00:00	111.00	111.00	111.00	111.0	111.00	12.050000	14.870000	111.0	57.369999	111
9	2010-03-01 01:00:00	1.48	0.18	0.51	111.0	111.00	16.780001	21.680000	111.0	78.660004	21
10	2010-03-01 01:00:00	111.00	0.22	111.00	111.0	111.00	21.450001	40.029999	111.0	111.000000	23

In [7]:
```
d=c[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3']]
```

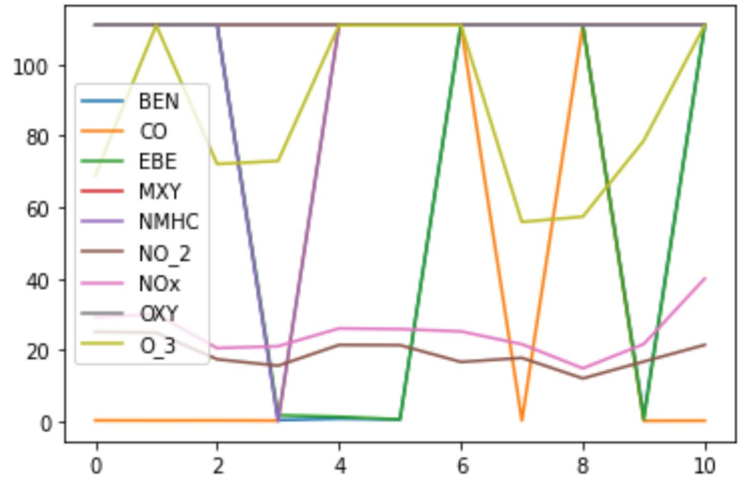
Out[7]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	111.00	0.29	111.00	111.0	111.00	25.090000	29.219999	111.0	68.930000
1	111.00	0.27	111.00	111.0	111.00	24.879999	30.040001	111.0	111.000000
2	111.00	0.28	111.00	111.0	111.00	17.410000	20.540001	111.0	72.120003
3	0.38	0.24	1.74	111.0	0.05	15.610000	21.080000	111.0	72.970001
4	0.79	111.00	1.32	111.0	111.00	21.430000	26.070000	111.0	111.000000
5	0.56	111.00	0.58	111.0	111.00	21.370001	25.870001	111.0	111.000000
6	111.00	111.00	111.00	111.0	111.00	16.660000	25.230000	111.0	111.000000
7	111.00	0.23	111.00	111.0	111.00	17.799999	21.639999	111.0	55.880001
8	111.00	111.00	111.00	111.0	111.00	12.050000	14.870000	111.0	57.369999
9	1.48	0.18	0.51	111.0	111.00	16.780001	21.680000	111.0	78.660004
10	111.00	0.22	111.00	111.0	111.00	21.450001	40.029999	111.0	111.000000

In [8]:

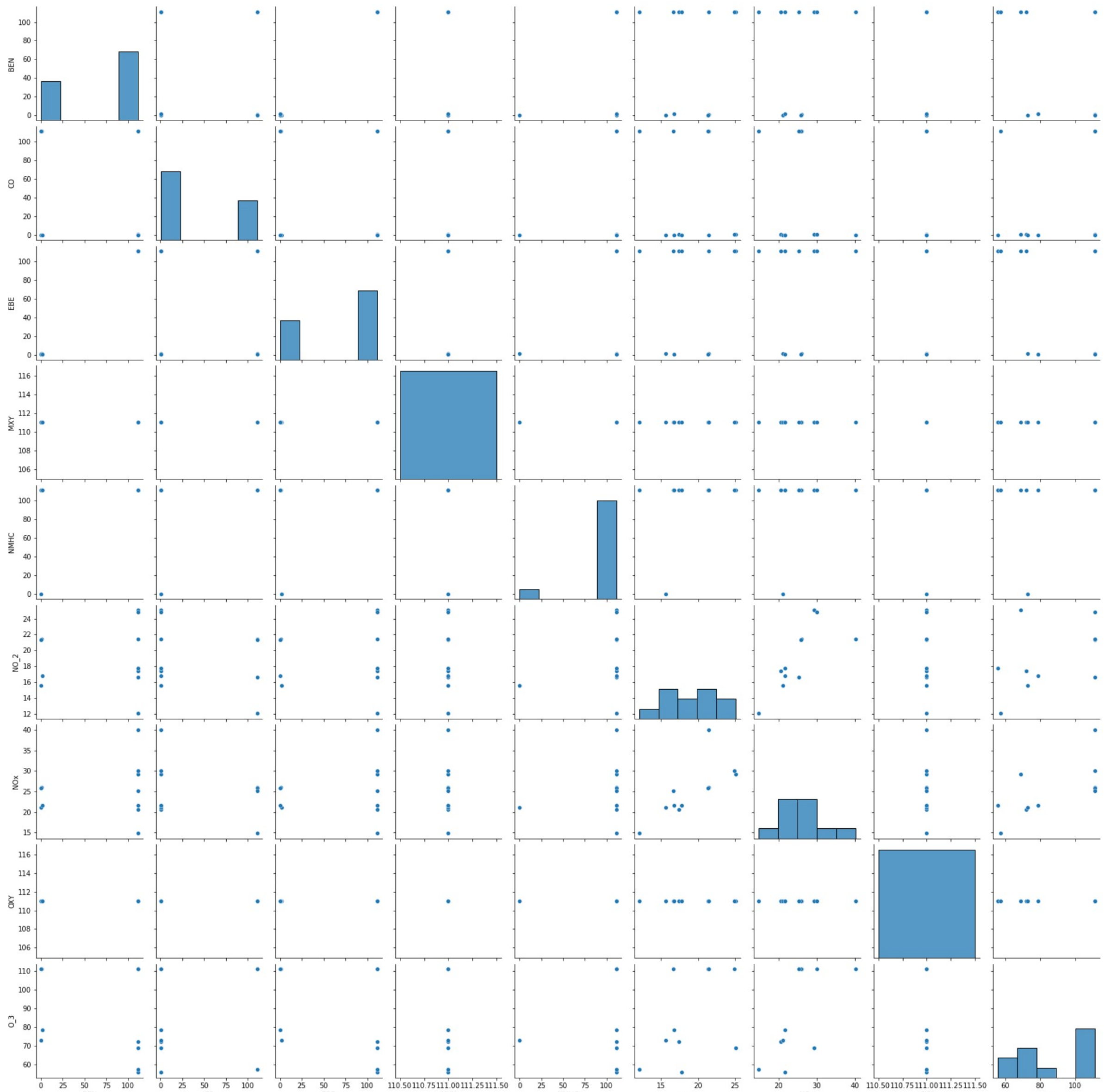
```
df.plot.line()
```

Out[8]: <AxesSubplot:>



In [9]:

Out[9]: `<seaborn.axisgrid.PairGrid at 0x2cfb0147460>`



In [10]:
```
x=d[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY']]
```

In [11]:
```
from sklearn.model_selection import train_test_split
```

In [12]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

Out[12]: `LinearRegression()`

In [13]:

```
-3.197442310920451e-14
```

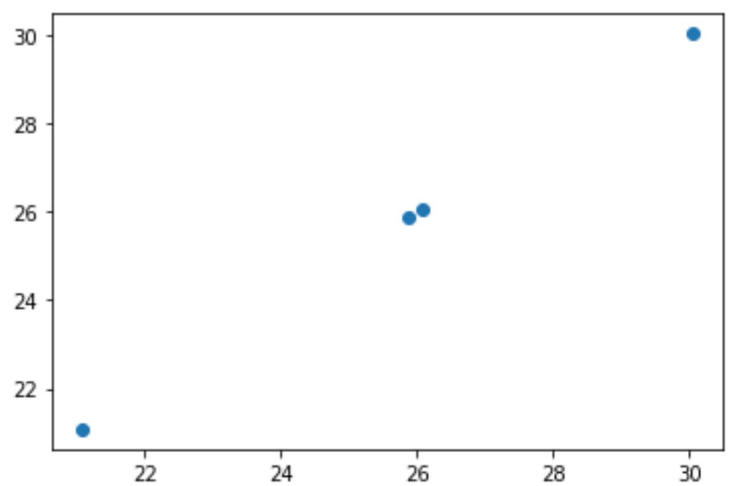
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	-9.267861e-16
CO	1.214265e-16
EBE	7.827542e-16
MXY	0.000000e+00
NMHC	0.000000e+00
NO_2	7.440214e-16
NOx	1.000000e+00
OXY	0.000000e+00

In [15]:
```
prediction=lr.predict(x_test)
```

Out[15]: <matplotlib.collections.PathCollection at 0x2cfb358ae20>



In [16]:

```
1.0
```

In [17]:

In [18]:
```
rr=Ridge(alpha=10)
```

Out[18]: Ridge(alpha=10)

In [19]:

Out[19]: 0.998143642386348

In [20]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[20]: Lasso(alpha=10)

In [21]:

Out[21]: 0.8953654254807312

In [22]:
```
a1=b.head(7000)
```

Out[22]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	2010-03-01 01:00:00	111.00	0.29	111.00	111.00	111.00	25.090000	29.219999	111.00	68.930000
1	2010-03-01 01:00:00	111.00	0.27	111.00	111.00	111.00	24.879999	30.040001	111.00	111.000000
2	2010-03-01 01:00:00	111.00	0.28	111.00	111.00	111.00	17.410000	20.540001	111.00	72.120003
3	2010-03-01 01:00:00	0.38	0.24	1.74	111.00	0.05	15.610000	21.080000	111.00	72.970001
4	2010-03-01 01:00:00	0.79	111.00	1.32	111.00	111.00	21.430000	26.070000	111.00	111.000000
...	...	...	...	...	...	...	...	...	...	...
6995	2010-03-13 06:00:00	0.69	0.26	0.47	0.53	0.23	40.490002	42.220001	0.84	22.170000
6996	2010-03-13 06:00:00	111.00	111.00	111.00	111.00	0.09	52.590000	66.339996	111.00	23.850000
6997	2010-03-13 06:00:00	111.00	111.00	111.00	111.00	111.00	41.950001	44.310001	111.00	111.000000
6998	2010-03-13 06:00:00	111.00	111.00	111.00	111.00	111.00	27.459999	30.540001	111.00	47.369999
6999	2010-03-13 06:00:00	111.00	111.00	111.00	111.00	111.00	36.830002	42.049999	111.00	111.000000

7000 rows × 17 columns

In [23]:
```
e=a1[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
```

In [24]:
```
f=e.iloc[:,0:14]
```

In [25]:

In [26]:
```
logr=LogisticRegression(max_iter=10000)
```

Out[26]: LogisticRegression(max_iter=10000)

In [27]:
```
from sklearn.model_selection import train_test_split
```

In [28]:

In [29]:
```
prediction=logr.predict(i)
```

```
[28079004]
```

In [30]:

Out[30]:
```
array([28079003, 28079004, 28079008, 28079011, 28079016, 28079017,
       28079018, 28079024, 28079027, 28079036, 28079038, 28079039,
       28079040, 28079047, 28079049, 28079050, 28079054, 28079055,
       28079056, 28079057, 28079058, 28079059, 28079060, 28079099],
      dtype=int64)
```

In [31]:

Out[31]:
```
5.006307963717743e-221
```

In [32]:

Out[32]:
```
0.9999999999999998
```

In [33]:

Out[33]:
```
0.8609523809523809
```

In [34]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
```

Out[34]:
```
ElasticNet()
```

In [35]:

```
[ 2.91057726e-04 -1.25597362e-03  5.05219264e-04  0.00000000e+00
  0.00000000e+00  0.00000000e+00  9.78104467e-01  0.00000000e+00]
```

In [36]:

```
0.5059785186362546
```

In [37]:
```
prediction=en.predict(x_test)
```

```
0.9978374272653505
```

In [38]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

Out[38]:
```
RandomForestClassifier()
```

In [39]:
```
parameters={'max_depth':[1,2,3,4,5],
 'min_samples_leaf':[5,10,15,20,25],
 'n_estimators':[10,20,30,40,50]
```

In [40]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="acc
```

Out[40]:
```
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 [41]:

Out[41]:
```
0.8322448979591837
```

In [42]:

In [43]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,50))
```

Out[43]:
```
[Text(2346.1363636363635, 2491.5, 'X[9] <= 0.652\ngini = 0.958\nsamples = 309
4\nvalue = [220, 164, 194, 189, 195, 195, 185, 216, 201, 194\n232, 211, 226,
190, 187, 225, 210, 198, 218, 225\n212, 211, 201, 201]'),
 Text(1268.1818181818182, 2038.5, 'X[2] <= -0.39\ngini = 0.916\nsamples = 154
0\nvalue = [0, 0, 194, 0, 0, 0, 185, 213, 0, 194, 232, 0\n226, 190, 0, 225,
0, 198, 0, 225, 0, 0, 201, 201]'),
 Text(760.9090909090909, 1585.5, 'X[1] <= -0.098\ngini = 0.832\nsamples = 76
7\nvalue = [0, 0, 194, 0, 0, 0, 185, 213, 0, 0, 232, 0, 0\n0, 0, 0, 0, 198,
0, 0, 0, 0, 0, 201]'),
 Text(405.8181818181818, 1132.5, 'X[3] <= -1.516\ngini = 0.749\nsamples = 49
1\nvalue = [0, 0, 194, 0, 0, 0, 185, 213, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 201]'),
 Text(202.9090909090909, 679.5, 'X[0] <= -1.56\ngini = 0.499\nsamples = 248\n
value = [0, 0, 0, 0, 0, 0, 0, 213, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 195]'),
 Text(101.45454545454545, 226.5, 'gini = 0.361\nsamples = 122\nvalue = [0, 0,
0, 0, 0, 0, 0, 158, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 49]'),
 Text(304.3636363636364, 226.5, 'gini = 0.398\nsamples = 126\nvalue = [0, 0,
0, 0, 0, 0, 0, 55, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 146]'),
```

From this observation I had observe that the RIDGE is a highest accuracy of 0.998143642386348

In [ ]: