In [1]:
```
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]:
```
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	2007-12-01 01:00:00	NaN	2.86	NaN	NaN	NaN	282.200012	1054.000000	NaN	4.030000	15
1	2007-12-01 01:00:00	NaN	1.82	NaN	NaN	NaN	86.419998	354.600006	NaN	3.260000	8
2	2007-12-01 01:00:00	NaN	1.47	NaN	NaN	NaN	94.639999	319.000000	NaN	5.310000	5
3	2007-12-01 01:00:00	NaN	1.64	NaN	NaN	NaN	127.900002	476.700012	NaN	4.500000	1(
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	1(
...	...	...	...	...	...	...	...	...	...	...	
225115	2007-03-01 00:00:00	0.30	0.45	1.00	0.30	0.26	8.690000	11.690000	1.00	42.209999	
225116	2007-03-01 00:00:00	NaN	0.16	NaN	NaN	NaN	46.820000	51.480000	NaN	22.150000	
225117	2007-03-01 00:00:00	0.24	NaN	0.20	NaN	0.09	51.259998	66.809998	NaN	18.540001	1
225118	2007-03-01 00:00:00	0.11	NaN	1.00	NaN	0.05	24.240000	36.930000	NaN	NaN	
225119	2007-03-01 00:00:00	0.53	0.40	1.00	1.70	0.12	32.360001	47.860001	1.37	24.150000	1

225120 rows × 17 columns

In [3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 225120 entries, 0 to 225119
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     225120 non-null  object 
 1   BEN      68885 non-null   float64
 2   CO       206748 non-null  float64
 3   EBE      68883 non-null   float64
 4   MXY      26061 non-null   float64
 5   NMHC     86883 non-null   float64
 6   NO_2     223985 non-null  float64
 7   NOx      223972 non-null  float64
 8   OXY      26062 non-null   float64
 9   O_3      211850 non-null  float64
 10  PM10     222588 non-null  float64
 11  PM25     68870 non-null   float64
 12  PXY      26062 non-null   float64
 13  SO_2     224372 non-null  float64
 14  TCH      87026 non-null   float64
 15  TOL      68845 non-null   float64
 16  station  225120 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 29.2+ MB
```

In [4]:
```
b=a.fillna(value=82)
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	2007-12-01 01:00:00	82.00	2.86	82.00	82.00	82.00	282.200012	1054.000000	82.00	4.030000
1	2007-12-01 01:00:00	82.00	1.82	82.00	82.00	82.00	86.419998	354.600006	82.00	3.260000
2	2007-12-01 01:00:00	82.00	1.47	82.00	82.00	82.00	94.639999	319.000000	82.00	5.310000
3	2007-12-01 01:00:00	82.00	1.64	82.00	82.00	82.00	127.900002	476.700012	82.00	4.500000
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999
...	...	...	...	...	...	...	...	...	...	..
225115	2007-03-01 00:00:00	0.30	0.45	1.00	0.30	0.26	8.690000	11.690000	1.00	42.209999
225116	2007-03-01 00:00:00	82.00	0.16	82.00	82.00	82.00	46.820000	51.480000	82.00	22.150000
225117	2007-03-01 00:00:00	0.24	82.00	0.20	82.00	0.09	51.259998	66.809998	82.00	18.540001
225118	2007-03-01 00:00:00	0.11	82.00	1.00	82.00	0.05	24.240000	36.930000	82.00	82.000000
225119	2007-03-01 00:00:00	0.53	0.40	1.00	1.70	0.12	32.360001	47.860001	1.37	24.150000

225120 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	2007-12-01 01:00:00	82.00	2.86	82.00	82.00	82.00	282.200012	1054.000000	82.00	4.030000	156
1	2007-12-01 01:00:00	82.00	1.82	82.00	82.00	82.00	86.419998	354.600006	82.00	3.260000	80
2	2007-12-01 01:00:00	82.00	1.47	82.00	82.00	82.00	94.639999	319.000000	82.00	5.310000	53
3	2007-12-01 01:00:00	82.00	1.64	82.00	82.00	82.00	127.900002	476.700012	82.00	4.500000	105
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	106
5	2007-12-01 01:00:00	82.00	1.35	82.00	82.00	0.56	115.300003	319.600006	82.00	9.880000	57
6	2007-12-01 01:00:00	5.54	1.87	4.65	82.00	0.75	165.100006	520.000000	82.00	4.780000	75
7	2007-12-01 01:00:00	82.00	1.57	82.00	82.00	82.00	97.830002	369.000000	82.00	4.870000	59
8	2007-12-01 01:00:00	82.00	0.70	82.00	82.00	82.00	107.699997	188.500000	82.00	4.560000	43
9	2007-12-01 01:00:00	82.00	1.48	82.00	82.00	0.69	152.500000	485.200012	82.00	8.230000	80
10	2007-12-01 01:00:00	82.00	1.87	82.00	82.00	82.00	113.500000	519.099976	82.00	4.380000	104

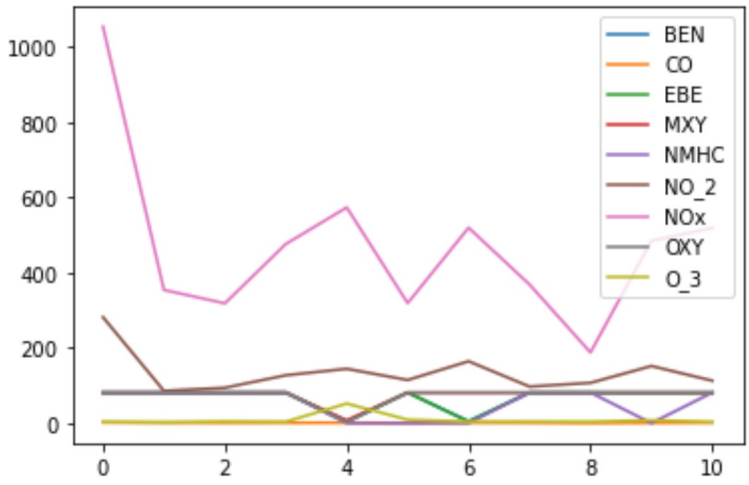
In [7]:
```
d=c[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3']]
```

Out[7]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	82.00	2.86	82.00	82.00	82.00	282.200012	1054.000000	82.00	4.030000
1	82.00	1.82	82.00	82.00	82.00	86.419998	354.600006	82.00	3.260000
2	82.00	1.47	82.00	82.00	82.00	94.639999	319.000000	82.00	5.310000
3	82.00	1.64	82.00	82.00	82.00	127.900002	476.700012	82.00	4.500000
4	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999
5	82.00	1.35	82.00	82.00	0.56	115.300003	319.600006	82.00	9.880000
6	5.54	1.87	4.65	82.00	0.75	165.100006	520.000000	82.00	4.780000
7	82.00	1.57	82.00	82.00	82.00	97.830002	369.000000	82.00	4.870000
8	82.00	0.70	82.00	82.00	82.00	107.699997	188.500000	82.00	4.560000
9	82.00	1.48	82.00	82.00	0.69	152.500000	485.200012	82.00	8.230000
10	82.00	1.87	82.00	82.00	82.00	113.500000	519.099976	82.00	4.380000

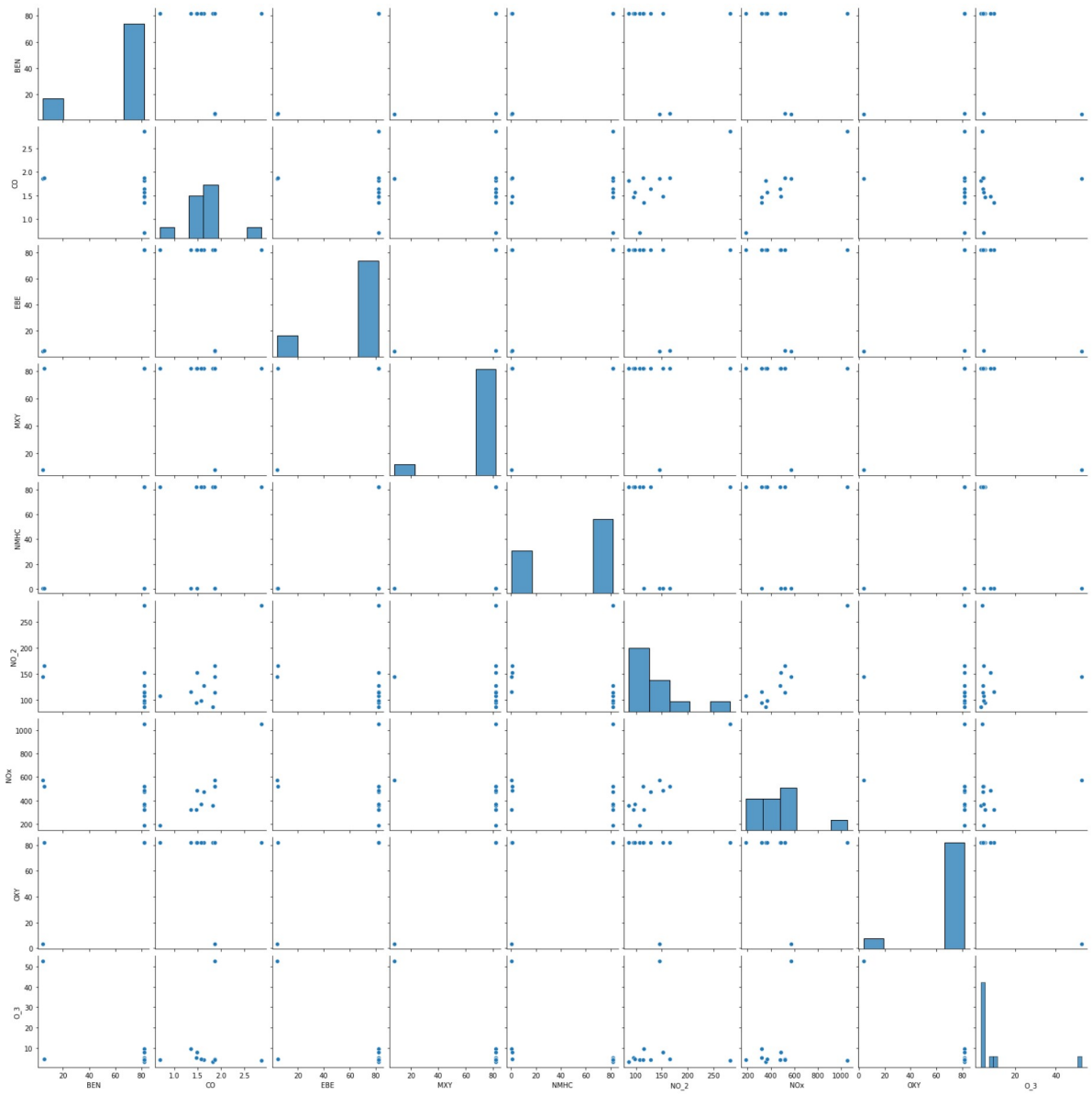
In [8]:

Out[8]: <AxesSubplot:>



In [9]:

Out[9]: <seaborn.axisgrid.PairGrid at 0x20704b35220>



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]:

```
-4.547473508864641e-13
```

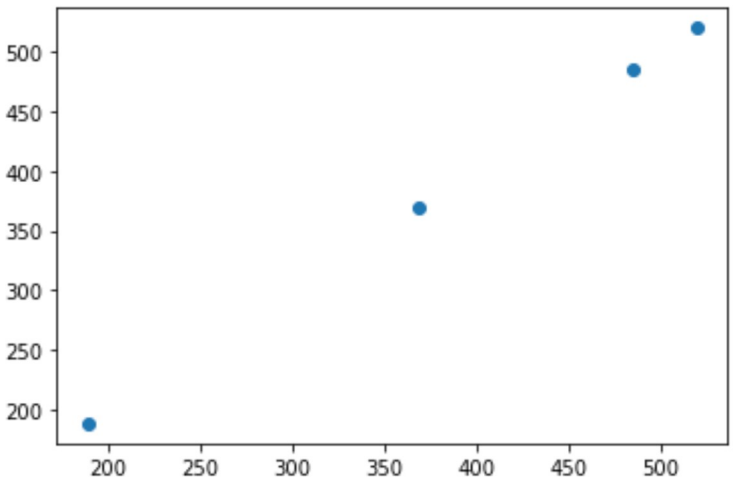
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	-3.983305e-16
CO	9.015748e-14
EBE	-2.838995e-16
MXY	-3.595983e-16
NMHC	1.794156e-15
NO_2	1.135338e-16
NOx	1.000000e+00
OXY	-3.416783e-16

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

Out[15]:
```
<matplotlib.collections.PathCollection at 0x2070991a1f0>
```



In [16]:

```
1.0
```

In [17]:

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

Out[18]:
```
Ridge(alpha=10)
```

In [19]:

Out[19]:
```
0.9999990139552668
```

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

Out[20]:
```
Lasso(alpha=10)
```

In [21]:

Out[21]: 0.9999999396703001

In [22]:
```
a1=b.head(6000)
```

Out[22]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2007-12-01 01:00:00	82.00	2.86	82.00	82.00	82.00	282.200012	1054.000000	82.00	4.030000	1
1	2007-12-01 01:00:00	82.00	1.82	82.00	82.00	82.00	86.419998	354.600006	82.00	3.260000	
2	2007-12-01 01:00:00	82.00	1.47	82.00	82.00	82.00	94.639999	319.000000	82.00	5.310000	
3	2007-12-01 01:00:00	82.00	1.64	82.00	82.00	82.00	127.900002	476.700012	82.00	4.500000	1
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	1
...	...	...	...	...	...	...	...	...	...	...	...
5995	2007-12-10 15:00:00	82.00	0.31	82.00	82.00	82.00	23.350000	28.080000	82.00	49.939999	
5996	2007-12-10 15:00:00	82.00	0.45	82.00	82.00	82.00	63.939999	112.000000	82.00	30.309999	
5997	2007-12-10 15:00:00	82.00	0.31	82.00	82.00	82.00	34.779999	52.369999	82.00	42.070000	
5998	2007-12-10 15:00:00	82.00	0.36	82.00	82.00	82.00	66.269997	112.099998	82.00	50.730000	
5999	2007-12-10 15:00:00	82.00	0.29	82.00	82.00	82.00	43.029999	66.809998	82.00	49.919998	

6000 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)
```

```
[28079038]
```

In [30]:

Out[30]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079018, 28079019, 28079021, 28079022, 28079023, 28079024,
       28079025, 28079026, 28079027, 28079036, 28079038, 28079039,
       28079040, 28079099], dtype=int64)
```

In [31]:

Out[31]:
```
3.9362863136010165e-59
```

In [32]:

Out[32]:
```
1.8650092005736393e-109
```

In [33]:

Out[33]:
```
0.48777777777777775
```

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

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

In [35]:

```
[-0.         0.        -0.        -0.         0.         0.
  0.9999824 -0.       ]
```

In [36]:

```
0.009092657542964844
```

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

```
0.9999999993967136
```

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]:
```
0.5407142857142857
```

In [42]:

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

Out[43]:
```
[Text(1935.9183673469388, 2491.5, 'X[11] <= -0.389\ngini = 0.961\nsamples = 2
643\nvalue = [153, 179, 159, 142, 192, 182, 147, 142, 150, 158\n164, 180, 16
9, 160, 144, 146, 148, 167, 184, 146\n186, 161, 170, 168, 151, 152]'),
 Text(842.6938775510205, 2038.5, 'X[10] <= -1.18\ngini = 0.947\nsamples = 122
1\nvalue = [10, 65, 40, 7, 4, 64, 93, 29, 36, 102, 66, 96\n28, 108, 68, 80, 1
06, 154, 154, 63, 170, 73, 28\n132, 87, 49]'),
 Text(364.40816326530614, 1585.5, 'X[5] <= -0.341\ngini = 0.407\nsamples = 13
2\nvalue = [0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 154, 0, 0, 0,
0, 0, 0, 0, 49]'),
 Text(182.20408163265307, 1132.5, 'X[8] <= -0.566\ngini = 0.482\nsamples = 10
0\nvalue = [0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 102, 0, 0, 0,
0, 0, 0, 0, 48]'),
 Text(91.10204081632654, 679.5, 'gini = 0.0\nsamples = 34\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 53, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(273.30612244897964, 679.5, 'X[12] <= -1.28\ngini = 0.56\nsamples = 66\n
value = [0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 49, 0, 0, 0, 0,
0, 0, 0, 48]'),
 Text(182.20408163265307, 226.5, 'gini = 0.499\nsamples = 43\nvalue = [0, 0,
0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 46, 0, 0, 0, 0, 0, 0, 0, 17]'),
```

From this observation I had observe that the ELASTICNET is a highest accuracy of 0.9999999993967136

In [ ]: