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
from sklearn.model_selection import train_test_split
```

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   2009-10-01 01:00:00   NaN  0.27   NaN   NaN   NaN  39.889999   48.150002   NaN  50.680000  18.2
     1   2009-10-01 01:00:00   NaN  0.22   NaN   NaN   NaN  21.230000   24.260000   NaN  55.880001  10.5
     2   2009-10-01 01:00:00   NaN  0.18   NaN   NaN   NaN  31.230000   34.880001   NaN  49.060001  25.1
     3   2009-10-01 01:00:00  0.95  0.33  1.43  2.68  0.25  55.180000   81.360001  1.57  36.669998  26.5
     4   2009-10-01 01:00:00   NaN  0.41   NaN   NaN  0.12  61.349998   76.260002   NaN  38.090000  23.7
   ...                   ...   ...   ...   ...   ...   ...        ...         ...   ...        ...
215683   2009-06-01 00:00:00  0.50  0.22  0.39  0.75  0.09  22.000000   24.510000  1.00  82.239998  10.8
215684   2009-06-01 00:00:00   NaN  0.31   NaN   NaN   NaN  76.110001  101.099998   NaN  41.220001   9.9
215685   2009-06-01 00:00:00  0.13   NaN  0.86   NaN  0.23  81.050003   99.849998   NaN  24.830000  12.4
215686   2009-06-01 00:00:00  0.21   NaN  2.96   NaN  0.10  72.419998   82.959999   NaN        NaN  13.0
215687   2009-06-01 00:00:00  0.37  0.32  0.99  1.36  0.14  54.290001   64.480003  1.06  56.919998  15.3

215688 rows × 17 columns
```

In [3]:

```
.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215688 entries, 0 to 215687
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     215688 non-null  object 
 1   BEN      60082 non-null   float64
 2   CO       190801 non-null  float64
 3   EBE      60081 non-null   float64
 4   MXY      24846 non-null   float64
 5   NMHC     74748 non-null   float64
 6   NO_2     214562 non-null  float64
 7   NOx      214565 non-null  float64
 8   OXY      24854 non-null   float64
 9   O_3      204482 non-null  float64
 10  PM10     196331 non-null  float64
 11  PM25     55822 non-null   float64
 12  PXY      24854 non-null   float64
 13  SO_2     212671 non-null  float64
 14  TCH      75213 non-null   float64
 15  TOL      59920 non-null   float64
 16  station  215688 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 28.0+ MB
```

In [4]:
```
b=a.fillna(value=98)
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	2009-10-01 01:00:00	98.00	0.27	98.00	98.00	98.00	39.889999	48.150002	98.00	50.680000
1	2009-10-01 01:00:00	98.00	0.22	98.00	98.00	98.00	21.230000	24.260000	98.00	55.880001
2	2009-10-01 01:00:00	98.00	0.18	98.00	98.00	98.00	31.230000	34.880001	98.00	49.060001
3	2009-10-01 01:00:00	0.95	0.33	1.43	2.68	0.25	55.180000	81.360001	1.57	36.669998
4	2009-10-01 01:00:00	98.00	0.41	98.00	98.00	0.12	61.349998	76.260002	98.00	38.090000
...	...	...	...	...	...	...	...	...	...	...
215683	2009-06-01 00:00:00	0.50	0.22	0.39	0.75	0.09	22.000000	24.510000	1.00	82.239998
215684	2009-06-01 00:00:00	98.00	0.31	98.00	98.00	98.00	76.110001	101.099998	98.00	41.220001
215685	2009-06-01 00:00:00	0.13	98.00	0.86	98.00	0.23	81.050003	99.849998	98.00	24.830000
215686	2009-06-01 00:00:00	0.21	98.00	2.96	98.00	0.10	72.419998	82.959999	98.00	98.000000
215687	2009-06-01 00:00:00	0.37	0.32	0.99	1.36	0.14	54.290001	64.480003	1.06	56.919998

215688 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)
```

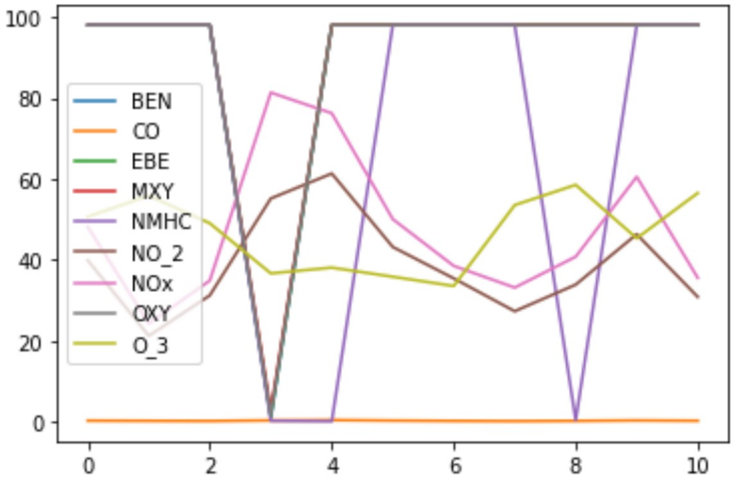
Out[6]:
```
                  date    BEN    CO    EBE    MXY   NMHC       NO_2        NOx    OXY        O_3     PM
 0   2009-10-01 01:00:00  98.00  0.27  98.00  98.00  98.00  39.889999  48.150002  98.00  50.680000  18.260
 1   2009-10-01 01:00:00  98.00  0.22  98.00  98.00  98.00  21.230000  24.260000  98.00  55.880001  10.580
 2   2009-10-01 01:00:00  98.00  0.18  98.00  98.00  98.00  31.230000  34.880001  98.00  49.060001  25.190
 3   2009-10-01 01:00:00   0.95  0.33   1.43   2.68   0.25  55.180000  81.360001   1.57  36.669998  26.530
 4   2009-10-01 01:00:00  98.00  0.41  98.00  98.00   0.12  61.349998  76.260002  98.00  38.090000  23.760
 5   2009-10-01 01:00:00  98.00  0.29  98.00  98.00  98.00  43.200001  50.080002  98.00  35.840000  21.870
 6   2009-10-01 01:00:00  98.00  0.20  98.00  98.00  98.00  35.430000  38.520000  98.00  33.549999  17.350
 7   2009-10-01 01:00:00  98.00  0.15  98.00  98.00  98.00  27.309999  33.150002  98.00  53.549999  16.520
 8   2009-10-01 01:00:00  98.00  0.21  98.00  98.00   0.39  33.889999  40.799999  98.00  58.549999  16.650
 9   2009-10-01 01:00:00  98.00  0.32  98.00  98.00  98.00  46.349998  60.540001  98.00  45.340000  15.160
10   2009-10-01 01:00:00  98.00  0.24  98.00  98.00  98.00  30.860001  35.590000  98.00  56.520000  14.420
```

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  98.00  0.27  98.00  98.00  98.00  39.889999  48.150002  98.00  50.680000
 1  98.00  0.22  98.00  98.00  98.00  21.230000  24.260000  98.00  55.880001
 2  98.00  0.18  98.00  98.00  98.00  31.230000  34.880001  98.00  49.060001
 3   0.95  0.33   1.43   2.68   0.25  55.180000  81.360001   1.57  36.669998
 4  98.00  0.41  98.00  98.00   0.12  61.349998  76.260002  98.00  38.090000
 5  98.00  0.29  98.00  98.00  98.00  43.200001  50.080002  98.00  35.840000
 6  98.00  0.20  98.00  98.00  98.00  35.430000  38.520000  98.00  33.549999
 7  98.00  0.15  98.00  98.00  98.00  27.309999  33.150002  98.00  53.549999
 8  98.00  0.21  98.00  98.00   0.39  33.889999  40.799999  98.00  58.549999
 9  98.00  0.32  98.00  98.00  98.00  46.349998  60.540001  98.00  45.340000
10  98.00  0.24  98.00  98.00  98.00  30.860001  35.590000  98.00  56.520000
```

In [8]:
```
b.plot.line()
```

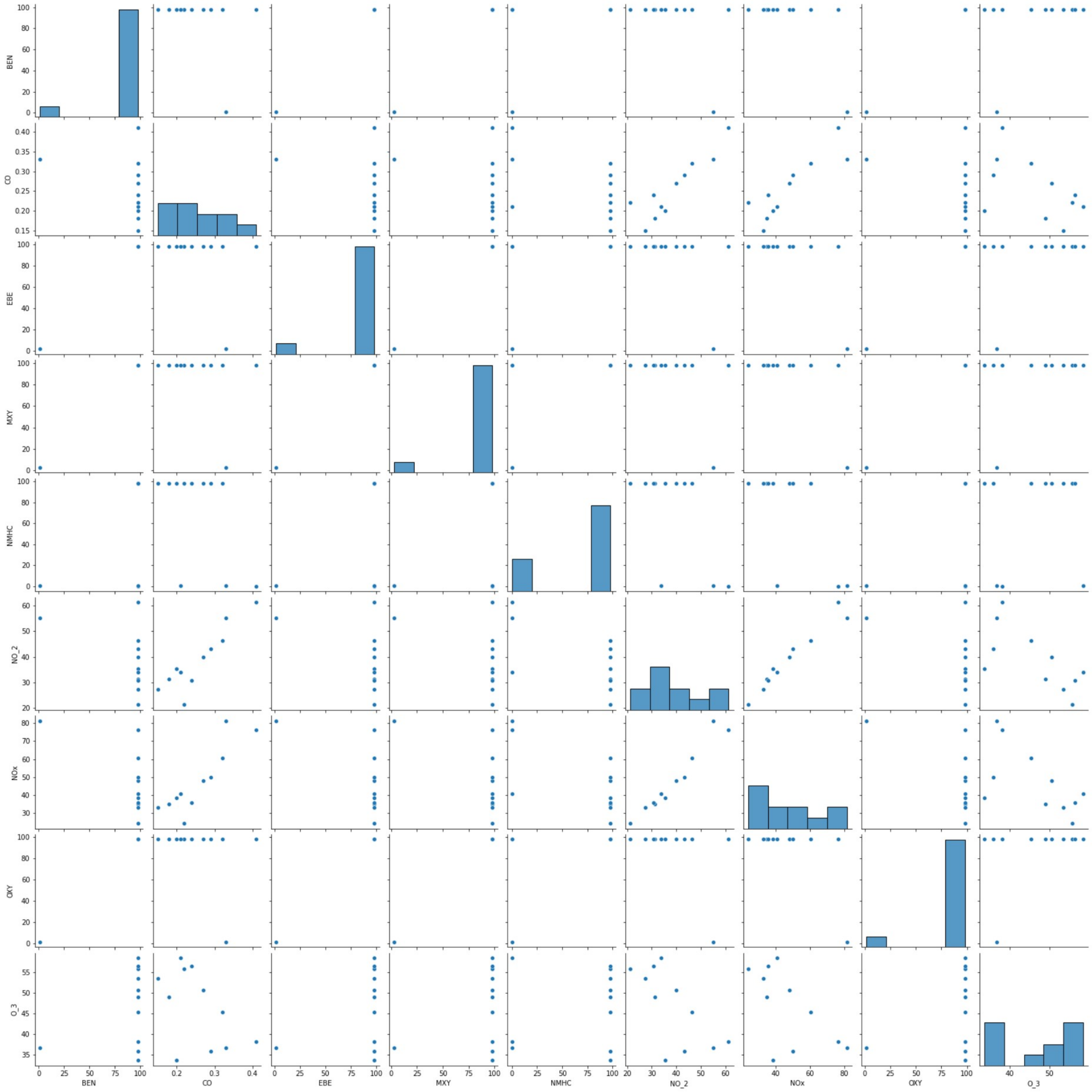
Out[8]:
```
<AxesSubplot:>
```



In [9]:

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x15228f57eb0>
```



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]:

```
-2.1316282072803006e-14
```

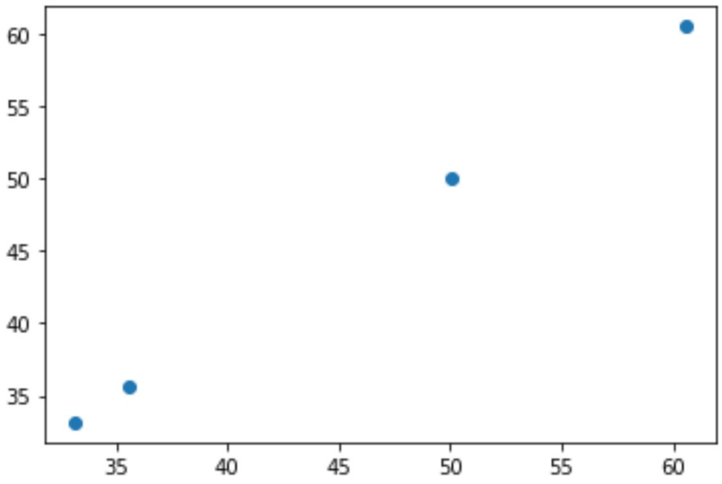
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	1.002284e-15
CO	-4.236772e-14
EBE	-1.779152e-16
MXY	-6.386554e-16
NMHC	-6.769435e-17
NO_2	1.006278e-15
NOx	1.000000e+00
OXY	-5.039871e-17

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

Out[15]:
```
<matplotlib.collections.PathCollection at 0x1522c531550>
```



In [16]:

```
1.0
```

In [17]:

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

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

In [19]:

Out[19]:
```
0.9932959728090598
```

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

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

In [21]:

Out[21]:
```
0.9980831728379033
```

In [22]:
```
a1=b.head(3000)
```

Out[22]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2009-10-01 01:00:00	98.00	0.27	98.00	98.00	98.00	39.889999	48.150002	98.00	50.680000	1
1	2009-10-01 01:00:00	98.00	0.22	98.00	98.00	98.00	21.230000	24.260000	98.00	55.880001	1
2	2009-10-01 01:00:00	98.00	0.18	98.00	98.00	98.00	31.230000	34.880001	98.00	49.060001	2
3	2009-10-01 01:00:00	0.95	0.33	1.43	2.68	0.25	55.180000	81.360001	1.57	36.669998	2
4	2009-10-01 01:00:00	98.00	0.41	98.00	98.00	0.12	61.349998	76.260002	98.00	38.090000	2
...	...	...	...	...	...	...	...	...	...	...	
2995	2009-10-06 00:00:00	1.10	0.75	1.16	2.85	0.66	128.300003	192.100006	0.92	0.600000	5
2996	2009-10-06 00:00:00	98.00	0.33	98.00	98.00	98.00	88.220001	108.199997	98.00	20.520000	4
2997	2009-10-06 00:00:00	1.10	98.00	0.39	98.00	0.42	124.400002	221.800003	98.00	6.190000	5
2998	2009-10-06 00:00:00	2.75	98.00	1.55	98.00	0.25	129.399994	166.000000	98.00	98.000000	2
2999	2009-10-06 00:00:00	2.23	0.83	2.36	7.03	0.44	126.300003	246.699997	3.18	8.110000	6

3000 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)
```
```
[28079021]
```

In [30]:

Out[30]:
```
array([28079003, 28079004, 28079006, 28079007, 28079008, 28079009,
       28079011, 28079012, 28079014, 28079016, 28079017, 28079018,
       28079019, 28079021, 28079022, 28079023, 28079024, 28079025,
       28079026, 28079027, 28079036, 28079038, 28079039, 28079040,
       28079099], dtype=int64)
```

In [31]:

Out[31]:
```
5.665341540178148e-99
```

In [32]:

Out[32]:
```
1.705432077416329e-33
```

In [33]:

Out[33]:
```
0.5888888888888889
```

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

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

In [35]:

```
[-0.          0.         -0.01381274 -0.0013857  -0.00329021  0.13429634
  0.88971371 -0.        ]
```

In [36]:

```
1.5507170991143298
```

In [37]:
```
prediction=en.predict(x_test)
```
```
0.9988450094456112
```

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]:
```
0.5780952380952381
```

In [42]:

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

Out[43]:
```
[Text(1923.3191489361702, 2491.5, 'X[10] <= -0.853\ngini = 0.96\nsamples = 13
21\nvalue = [87, 94, 80, 90, 92, 88, 81, 85, 71, 89, 72, 81\n90, 104, 87, 73,
65, 87, 88, 77, 76, 79, 90, 83\n91]'),
 Text(831.063829787234, 2038.5, 'X[7] <= -2.742\ngini = 0.661\nsamples = 152\
nvalue = [0, 0, 80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 65, 0, 0, 0, 0, 0,
0, 0, 91]'),
 Text(379.9148936170213, 1585.5, 'X[5] <= -0.393\ngini = 0.316\nsamples = 52\
nvalue = [0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 58, 0, 0, 0, 0, 0,
0, 0, 6]'),
 Text(189.95744680851064, 1132.5, 'X[2] <= -1.637\ngini = 0.166\nsamples = 3
9\nvalue = [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 51, 0, 0, 0, 0,
0, 0, 0, 2]'),
 Text(94.97872340425532, 679.5, 'gini = 0.0\nsamples = 27\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 38, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(284.93617021276594, 679.5, 'X[6] <= -0.816\ngini = 0.438\nsamples = 12\
nvalue = [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 13, 0, 0, 0, 0, 0,
0, 0, 2]'),
 Text(189.95744680851064, 226.5, 'gini = 0.625\nsamples = 7\nvalue = [0, 0,
2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 2]'),
```

From this observation I had observe that the ELASTICNET is a highest accuracy of 0.9988450094456112

In [ ]: