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	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL
0	2013-11-01 01:00:00	NaN	0.6	NaN	NaN	135.0	74.0	NaN	NaN	NaN	7.0	NaN	NaN
1	2013-11-01 01:00:00	1.5	0.5	1.3	NaN	71.0	83.0	2.0	23.0	16.0	12.0	NaN	8.3
2	2013-11-01 01:00:00	3.9	NaN	2.8	NaN	49.0	70.0	NaN	NaN	NaN	NaN	NaN	9.0
3	2013-11-01 01:00:00	NaN	0.5	NaN	NaN	82.0	87.0	3.0	NaN	NaN	NaN	NaN	NaN
4	2013-11-01 01:00:00	NaN	NaN	NaN	NaN	242.0	111.0	2.0	NaN	NaN	12.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
209875	2013-03-01 00:00:00	NaN	0.4	NaN	NaN	8.0	39.0	52.0	NaN	NaN	NaN	NaN	NaN
209876	2013-03-01 00:00:00	NaN	0.4	NaN	NaN	1.0	11.0	NaN	6.0	NaN	2.0	NaN	NaN
209877	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	2.0	4.0	75.0	NaN	NaN	NaN	NaN	NaN
209878	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	2.0	11.0	52.0	NaN	NaN	NaN	NaN	NaN
209879	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	1.0	10.0	75.0	3.0	NaN	NaN	NaN	NaN

209880 rows × 14 columns

In [3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209880 entries, 0 to 209879
Data columns (total 14 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     209880 non-null  object 
 1   BEN      50462 non-null   float64
 2   CO       87018 non-null   float64
 3   EBE      50463 non-null   float64
 4   NMHC     25935 non-null   float64
 5   NO       209108 non-null  float64
 6   NO_2     209108 non-null  float64
 7   O_3      121858 non-null  float64
 8   PM10     104339 non-null  float64
 9   PM25     51980 non-null   float64
 10  SO_2     86970 non-null   float64
 11  TCH      25935 non-null   float64
 12  TOL      50317 non-null   float64
 13  station  209880 non-null  int64  
dtypes: float64(12), int64(1), object(1)
memory usage: 22.4+ MB
```

In [4]:
```
b=a.fillna(value=122)
```

Out[4]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH
0	2013-11-01 01:00:00	122.0	0.6	122.0	122.0	135.0	74.0	122.0	122.0	122.0	7.0	122.0
1	2013-11-01 01:00:00	1.5	0.5	1.3	122.0	71.0	83.0	2.0	23.0	16.0	12.0	122.0
2	2013-11-01 01:00:00	3.9	122.0	2.8	122.0	49.0	70.0	122.0	122.0	122.0	122.0	122.0
3	2013-11-01 01:00:00	122.0	0.5	122.0	122.0	82.0	87.0	3.0	122.0	122.0	122.0	122.0
4	2013-11-01 01:00:00	122.0	122.0	122.0	122.0	242.0	111.0	2.0	122.0	122.0	12.0	122.0
...	...	...	...	...	...	...	...	...	...	...	...	...
209875	2013-03-01 00:00:00	122.0	0.4	122.0	122.0	8.0	39.0	52.0	122.0	122.0	122.0	122.0
209876	2013-03-01 00:00:00	122.0	0.4	122.0	122.0	1.0	11.0	122.0	6.0	122.0	2.0	122.0
209877	2013-03-01 00:00:00	122.0	122.0	122.0	122.0	2.0	4.0	75.0	122.0	122.0	122.0	122.0
209878	2013-03-01 00:00:00	122.0	122.0	122.0	122.0	2.0	11.0	52.0	122.0	122.0	122.0	122.0
209879	2013-03-01 00:00:00	122.0	122.0	122.0	122.0	1.0	10.0	75.0	3.0	122.0	122.0	122.0

209880 rows × 14 columns

In [5]:

Out[5]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'O_3', 'PM10', 'PM25
',
       'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

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

Out[6]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	T
0	2013-11-01 01:00:00	122.0	0.6	122.0	122.00	135.0	74.0	122.0	122.0	122.0	7.0	122.00	12
1	2013-11-01 01:00:00	1.5	0.5	1.3	122.00	71.0	83.0	2.0	23.0	16.0	12.0	122.00	8
2	2013-11-01 01:00:00	3.9	122.0	2.8	122.00	49.0	70.0	122.0	122.0	122.0	122.0	122.00	9
3	2013-11-01 01:00:00	122.0	0.5	122.0	122.00	82.0	87.0	3.0	122.0	122.0	122.0	122.00	12
4	2013-11-01 01:00:00	122.0	122.0	122.0	122.00	242.0	111.0	2.0	122.0	122.0	12.0	122.00	12
5	2013-11-01 01:00:00	1.0	0.6	0.8	122.00	70.0	70.0	2.0	24.0	122.0	6.0	122.00	5
6	2013-11-01 01:00:00	122.0	0.4	122.0	0.29	51.0	80.0	5.0	23.0	14.0	4.0	1.44	12
7	2013-11-01 01:00:00	122.0	122.0	122.0	0.23	29.0	60.0	4.0	122.0	122.0	122.0	1.51	12
8	2013-11-01 01:00:00	122.0	1.0	122.0	122.00	165.0	107.0	2.0	122.0	122.0	11.0	122.00	12
9	2013-11-01 01:00:00	122.0	0.6	122.0	122.00	63.0	93.0	122.0	11.0	122.0	8.0	122.00	12
10	2013-11-01 01:00:00	1.4	122.0	1.4	122.00	68.0	84.0	122.0	26.0	11.0	6.0	122.00	7

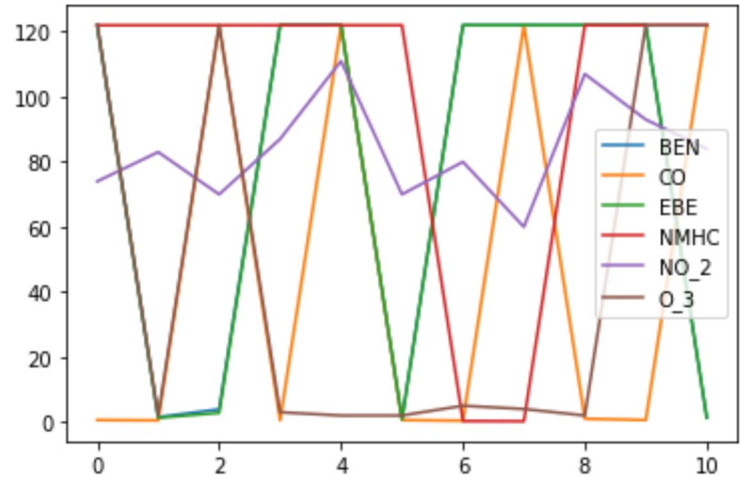
In [7]:
```
d=c[['BEN','CO','EBE','NMHC','NO_2','O_3']]
```

Out[7]:

	BEN	CO	EBE	NMHC	NO_2	O_3
0	122.0	0.6	122.0	122.00	74.0	122.0
1	1.5	0.5	1.3	122.00	83.0	2.0
2	3.9	122.0	2.8	122.00	70.0	122.0
3	122.0	0.5	122.0	122.00	87.0	3.0
4	122.0	122.0	122.0	122.00	111.0	2.0
5	1.0	0.6	0.8	122.00	70.0	2.0
6	122.0	0.4	122.0	0.29	80.0	5.0
7	122.0	122.0	122.0	0.23	60.0	4.0
8	122.0	1.0	122.0	122.00	107.0	2.0
9	122.0	0.6	122.0	122.00	93.0	122.0
10	1.4	122.0	1.4	122.00	84.0	122.0

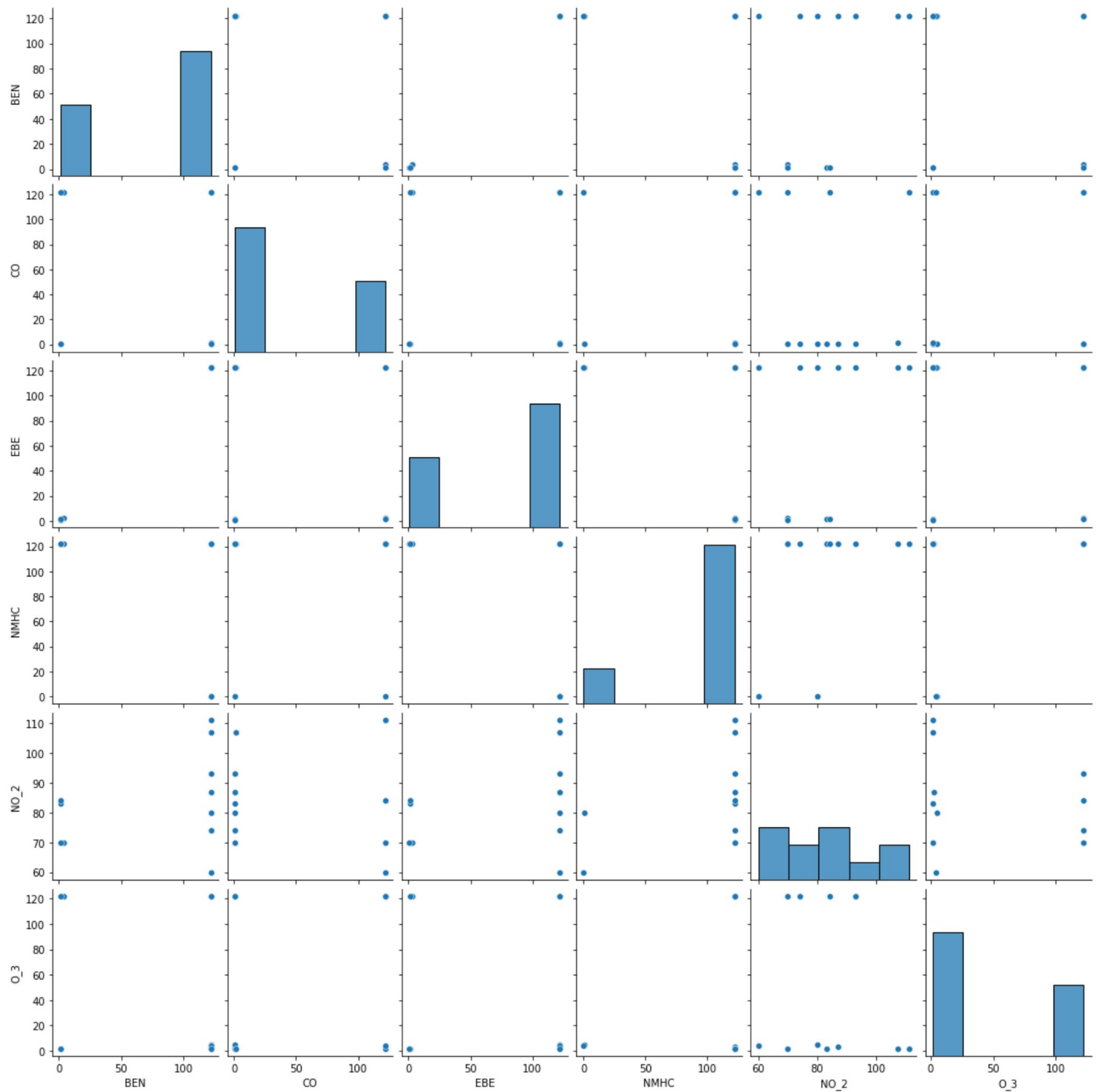
In [8]:
```
df.plot.line()
```

Out[8]: <AxesSubplot:>



In [9]:

Out[9]: <seaborn.axisgrid.PairGrid at 0x23d42171820>



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

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]:

8.526512829121202e-14

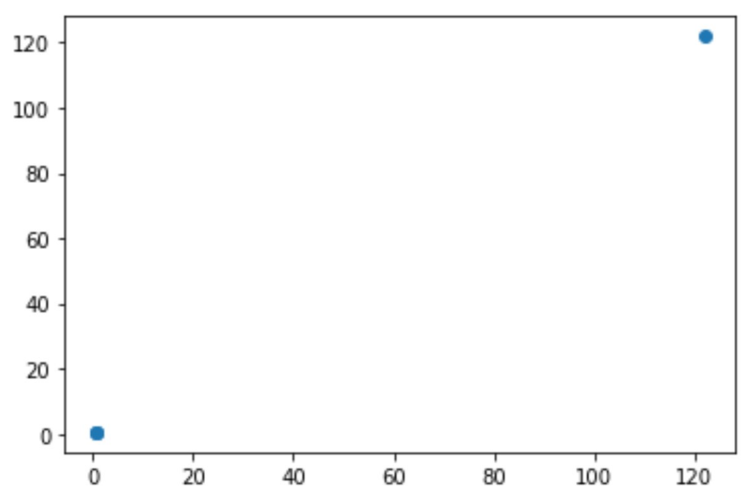
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	0.000000e+00
CO	1.000000e+00
EBE	-1.256790e-16
NMHC	4.884001e-18
NO_2	-7.198288e-16

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

Out[15]: <matplotlib.collections.PathCollection at 0x23d45204a30>



In [16]:

```
1.0
```

In [17]:

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

Out[18]: Ridge(alpha=10)

In [19]:

Out[19]: 0.9999992236551154

In [20]:
```
la=Lasso(alpha=10)
```

Out[20]: Lasso(alpha=10)

In [21]:

Out[21]: 0.9999910068713143

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

Out[22]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	T
0	2013-11-01 01:00:00	122.0	0.6	122.0	122.0	135.0	74.0	122.0	122.0	122.0	7.0	122.0	12
1	2013-11-01 01:00:00	1.5	0.5	1.3	122.0	71.0	83.0	2.0	23.0	16.0	12.0	122.0	
2	2013-11-01 01:00:00	3.9	122.0	2.8	122.0	49.0	70.0	122.0	122.0	122.0	122.0	122.0	
3	2013-11-01 01:00:00	122.0	0.5	122.0	122.0	82.0	87.0	3.0	122.0	122.0	122.0	122.0	12
4	2013-11-01 01:00:00	122.0	122.0	122.0	122.0	242.0	111.0	2.0	122.0	122.0	12.0	122.0	12
...	...	...	...	...	...	...	...	...	...	...	...	...	
5995	2013-11-11 10:00:00	122.0	1.1	122.0	122.0	202.0	93.0	7.0	122.0	122.0	122.0	122.0	12
5996	2013-11-11 10:00:00	122.0	0.8	122.0	122.0	170.0	100.0	122.0	44.0	122.0	12.0	122.0	12
5997	2013-11-11 10:00:00	122.0	122.0	122.0	122.0	14.0	27.0	4.0	122.0	122.0	122.0	122.0	12
5998	2013-11-11 10:00:00	122.0	122.0	122.0	122.0	78.0	50.0	9.0	122.0	122.0	122.0	122.0	12
5999	2013-11-11 10:00:00	122.0	122.0	122.0	122.0	181.0	102.0	9.0	53.0	122.0	122.0	122.0	12

6000 rows × 14 columns

In [23]:
```
e=a1[['BEN', 'CO', 'EBE','NMHC', 'NO_2','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)
```

```
[28079050]
```

In [30]:

Out[30]:

```
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 [31]:

Out[31]:

```
0.0
```

In [32]:

Out[32]:

```
0.0
```

In [33]:

Out[33]:

```
0.9511111111111111
```

In [34]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
```

Out[34]:

```
ElasticNet()
```

In [35]:

```
[-0.          0.99972286 -0.          -0.          -0.        ]
```

In [36]:

```
0.014589219172414403
```

In [37]:

```
prediction=en.predict(x_test)
```

```
0.9999999100936346
```

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]:

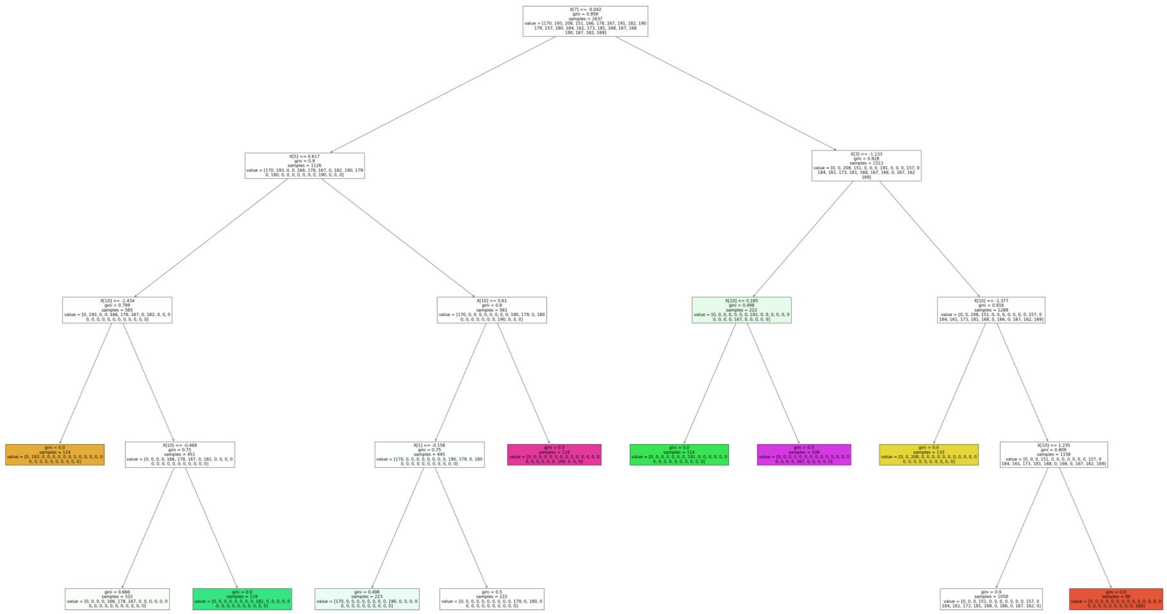
Out[41]:
```
0.9992857142857143
```

In [42]:

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

Out[43]:
```
[Text(2232.0, 2446.2, 'X[7] <= -0.042\ngini = 0.958\nsamples = 2637\nvalue =
[170, 193, 208, 151, 166, 178, 167, 191, 182, 190\n179, 157, 180, 184, 161, 1
73, 181, 168, 167, 166\n190, 167, 162, 169]'),
 Text(1174.7368421052631, 1902.6, 'X[5] <= 0.617\ngini = 0.9\nsamples = 1126\
nvalue = [170, 193, 0, 0, 166, 178, 167, 0, 182, 190, 179\n0, 180, 0, 0, 0,
0, 0, 0, 0, 190, 0, 0, 0]'),
 Text(469.89473684210526, 1359.0, 'X[10] <= -1.434\ngini = 0.799\nsamples = 5
65\nvalue = [0, 193, 0, 0, 166, 178, 167, 0, 182, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0]'),
 Text(234.94736842105263, 815.3999999999999, 'gini = 0.0\nsamples = 114\nvalu
e = [0, 193, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(704.8421052631579, 815.3999999999999, 'X[10] <= -0.468\ngini = 0.75\nsa
mples = 451\nvalue = [0, 0, 0, 0, 166, 178, 167, 0, 182, 0, 0, 0, 0\n0, 0,
0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(469.89473684210526, 271.7999999999997, 'gini = 0.666\nsamples = 332\nva
lue = [0, 0, 0, 0, 166, 178, 167, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(939.7894736842105, 271.7999999999997, 'gini = 0.0\nsamples = 119\nvalue
= [0, 0, 0, 0, 0, 0, 0, 0, 182, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1879.578947368421, 1359.0, 'X[10] <= 0.61\ngini = 0.8\nsamples = 561\nv
alue = [170, 0, 0, 0, 0, 0, 0, 0, 0, 190, 179, 0, 180\n0, 0, 0, 0, 0, 0, 0, 1
90, 0, 0, 0]'),
 Text(1644.6315789473683, 815.3999999999999, 'X[1] <= -0.158\ngini = 0.75\nsa
mples = 445\nvalue = [170, 0, 0, 0, 0, 0, 0, 0, 0, 190, 179, 0, 180\n0, 0,
0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1409.6842105263158, 271.7999999999997, 'gini = 0.498\nsamples = 223\nva
lue = [170, 0, 0, 0, 0, 0, 0, 0, 0, 190, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(1879.578947368421, 271.7999999999997, 'gini = 0.5\nsamples = 222\nvalue
= [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 179, 0, 180, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(2114.5263157894738, 815.3999999999999, 'gini = 0.0\nsamples = 116\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 190, 0, 0,
0]'),
 Text(3289.2631578947367, 1902.6, 'X[3] <= -1.133\ngini = 0.928\nsamples = 15
11\nvalue = [0, 0, 208, 151, 0, 0, 0, 191, 0, 0, 0, 157, 0\n184, 161, 173, 18
1, 168, 167, 166, 0, 167, 162\n169]'),
 Text(2819.3684210526317, 1359.0, 'X[10] <= 0.185\ngini = 0.498\nsamples = 22
2\nvalue = [0, 0, 0, 0, 0, 0, 0, 191, 0, 0, 0, 0, 0, 0\n0, 0, 0, 167, 0,
0, 0, 0, 0]'),
 Text(2584.4210526315787, 815.3999999999999, 'gini = 0.0\nsamples = 114\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 191, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(3054.315789473684, 815.3999999999999, 'gini = 0.0\nsamples = 108\nvalue
= [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 167, 0, 0, 0, 0,
0]'),
 Text(3759.157894736842, 1359.0, 'X[10] <= -1.377\ngini = 0.916\nsamples = 12
89\nvalue = [0, 0, 208, 151, 0, 0, 0, 0, 0, 0, 0, 157, 0\n184, 161, 173, 181,
168, 0, 166, 0, 167, 162, 169]'),
 Text(3524.2105263157896, 815.3999999999999, 'gini = 0.0\nsamples = 133\nvalu
e = [0, 0, 208, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
```

```
0]'),
 Text(3994.1052631578946, 815.3999999999999, 'X[10] <= 1.235\ngini = 0.909\nsamples = 1156\nvalue = [0, 0, 0, 151, 0, 0, 0, 0, 0, 0, 0, 157, 0\n184, 161, 173, 181, 168, 0, 166, 0, 167, 162, 169]'),
 Text(3759.157894736842, 271.7999999999997, 'gini = 0.9\nsamples = 1058\nvalue = [0, 0, 0, 151, 0, 0, 0, 0, 0, 0, 0, 157, 0\n184, 161, 173, 181, 168, 0, 166, 0, 167, 162, 0]'),
 Text(4229.0526315789475, 271.7999999999997, 'gini = 0.0\nsamples = 98\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 169]')]
```



From this observation I had observe that the ELASTICNET is a highest accuracy of 0.9999999100936346

In [ ]: