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	2011-11-01 01:00:00	NaN	1.0	NaN	NaN	154.0	84.0	NaN	NaN	NaN	6.0	NaN	NaN
1	2011-11-01 01:00:00	2.5	0.4	3.5	0.26	68.0	92.0	3.0	40.0	24.0	9.0	1.54	8.7
2	2011-11-01 01:00:00	2.9	NaN	3.8	NaN	96.0	99.0	NaN	NaN	NaN	NaN	NaN	7.2
3	2011-11-01 01:00:00	NaN	0.6	NaN	NaN	60.0	83.0	2.0	NaN	NaN	NaN	NaN	NaN
4	2011-11-01 01:00:00	NaN	NaN	NaN	NaN	44.0	62.0	3.0	NaN	NaN	3.0	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
209923	2011-09-01 00:00:00	NaN	0.2	NaN	NaN	5.0	19.0	44.0	NaN	NaN	NaN	NaN	NaN
209924	2011-09-01 00:00:00	NaN	0.1	NaN	NaN	6.0	29.0	NaN	11.0	NaN	7.0	NaN	NaN
209925	2011-09-01 00:00:00	NaN	NaN	NaN	0.23	1.0	21.0	28.0	NaN	NaN	NaN	1.44	NaN
209926	2011-09-01 00:00:00	NaN	NaN	NaN	NaN	3.0	15.0	48.0	NaN	NaN	NaN	NaN	NaN
209927	2011-09-01 00:00:00	NaN	NaN	NaN	NaN	4.0	33.0	38.0	13.0	NaN	NaN	NaN	NaN

209928 rows × 14 columns

In [3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209928 entries, 0 to 209927
Data columns (total 14 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     209928 non-null  object 
 1   BEN      51393 non-null   float64
 2   CO       87127 non-null   float64
 3   EBE      51350 non-null   float64
 4   NMHC     43517 non-null   float64
 5   NO       208954 non-null  float64
 6   NO_2     208973 non-null  float64
 7   O_3      122049 non-null  float64
 8   PM10     103743 non-null  float64
 9   PM25     51079 non-null   float64
 10  SO_2     87131 non-null   float64
 11  TCH      43519 non-null   float64
 12  TOL      51175 non-null   float64
 13  station  209928 non-null  int64  
dtypes: float64(12), int64(1), object(1)
memory usage: 22.4+ MB
```

In [4]:
```
b=a.fillna(value=226)
```

Out[4]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH
0	2011-11-01 01:00:00	226.0	1.0	226.0	226.00	154.0	84.0	226.0	226.0	226.0	6.0	226.00
1	2011-11-01 01:00:00	2.5	0.4	3.5	0.26	68.0	92.0	3.0	40.0	24.0	9.0	1.54
2	2011-11-01 01:00:00	2.9	226.0	3.8	226.00	96.0	99.0	226.0	226.0	226.0	226.0	226.00
3	2011-11-01 01:00:00	226.0	0.6	226.0	226.00	60.0	83.0	2.0	226.0	226.0	226.0	226.00
4	2011-11-01 01:00:00	226.0	226.0	226.0	226.00	44.0	62.0	3.0	226.0	226.0	3.0	226.00
...	...	...	...	...	...	...	...	...	...	...	...	...
209923	2011-09-01 00:00:00	226.0	0.2	226.0	226.00	5.0	19.0	44.0	226.0	226.0	226.0	226.00
209924	2011-09-01 00:00:00	226.0	0.1	226.0	226.00	6.0	29.0	226.0	11.0	226.0	7.0	226.00
209925	2011-09-01 00:00:00	226.0	226.0	226.0	0.23	1.0	21.0	28.0	226.0	226.0	226.0	1.44
209926	2011-09-01 00:00:00	226.0	226.0	226.0	226.00	3.0	15.0	48.0	226.0	226.0	226.0	226.00
209927	2011-09-01 00:00:00	226.0	226.0	226.0	226.00	4.0	33.0	38.0	13.0	226.0	226.0	226.00

209928 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	TC
0	2011-11-01 01:00:00	226.0	1.0	226.0	226.00	154.0	84.0	226.0	226.0	226.0	6.0	226.00	226
1	2011-11-01 01:00:00	2.5	0.4	3.5	0.26	68.0	92.0	3.0	40.0	24.0	9.0	1.54	8
2	2011-11-01 01:00:00	2.9	226.0	3.8	226.00	96.0	99.0	226.0	226.0	226.0	226.0	226.00	7
3	2011-11-01 01:00:00	226.0	0.6	226.0	226.00	60.0	83.0	2.0	226.0	226.0	226.0	226.00	226
4	2011-11-01 01:00:00	226.0	226.0	226.0	226.00	44.0	62.0	3.0	226.0	226.0	3.0	226.00	226
5	2011-11-01 01:00:00	0.5	0.8	0.3	226.00	102.0	75.0	2.0	35.0	226.0	5.0	226.00	4
6	2011-11-01 01:00:00	0.7	0.3	1.1	0.16	17.0	66.0	7.0	22.0	16.0	2.0	1.36	1
7	2011-11-01 01:00:00	226.0	226.0	226.0	0.36	83.0	78.0	6.0	226.0	226.0	226.0	1.80	226
8	2011-11-01 01:00:00	226.0	0.7	226.0	226.00	80.0	91.0	5.0	226.0	226.0	8.0	226.00	226
9	2011-11-01 01:00:00	226.0	0.6	226.0	226.00	63.0	71.0	226.0	33.0	226.0	6.0	226.00	226
10	2011-11-01 01:00:00	0.3	226.0	1.4	226.00	77.0	81.0	226.0	41.0	23.0	5.0	226.00	6

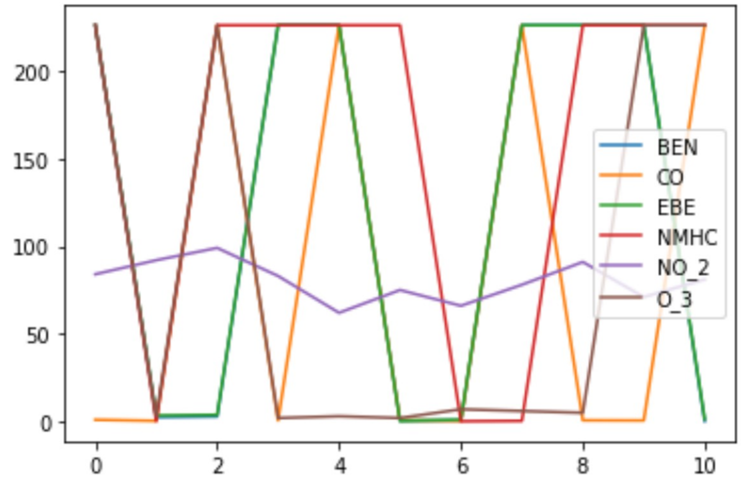
In [16]:
```
d=c[['BEN','CO','EBE','NMHC','NO_2','O_3']]
```

Out[16]:

	BEN	CO	EBE	NMHC	NO_2	O_3
0	226.0	1.0	226.0	226.00	84.0	226.0
1	2.5	0.4	3.5	0.26	92.0	3.0
2	2.9	226.0	3.8	226.00	99.0	226.0
3	226.0	0.6	226.0	226.00	83.0	2.0
4	226.0	226.0	226.0	226.00	62.0	3.0
5	0.5	0.8	0.3	226.00	75.0	2.0
6	0.7	0.3	1.1	0.16	66.0	7.0
7	226.0	226.0	226.0	0.36	78.0	6.0
8	226.0	0.7	226.0	226.00	91.0	5.0
9	226.0	0.6	226.0	226.00	71.0	226.0
10	0.3	226.0	1.4	226.00	81.0	226.0

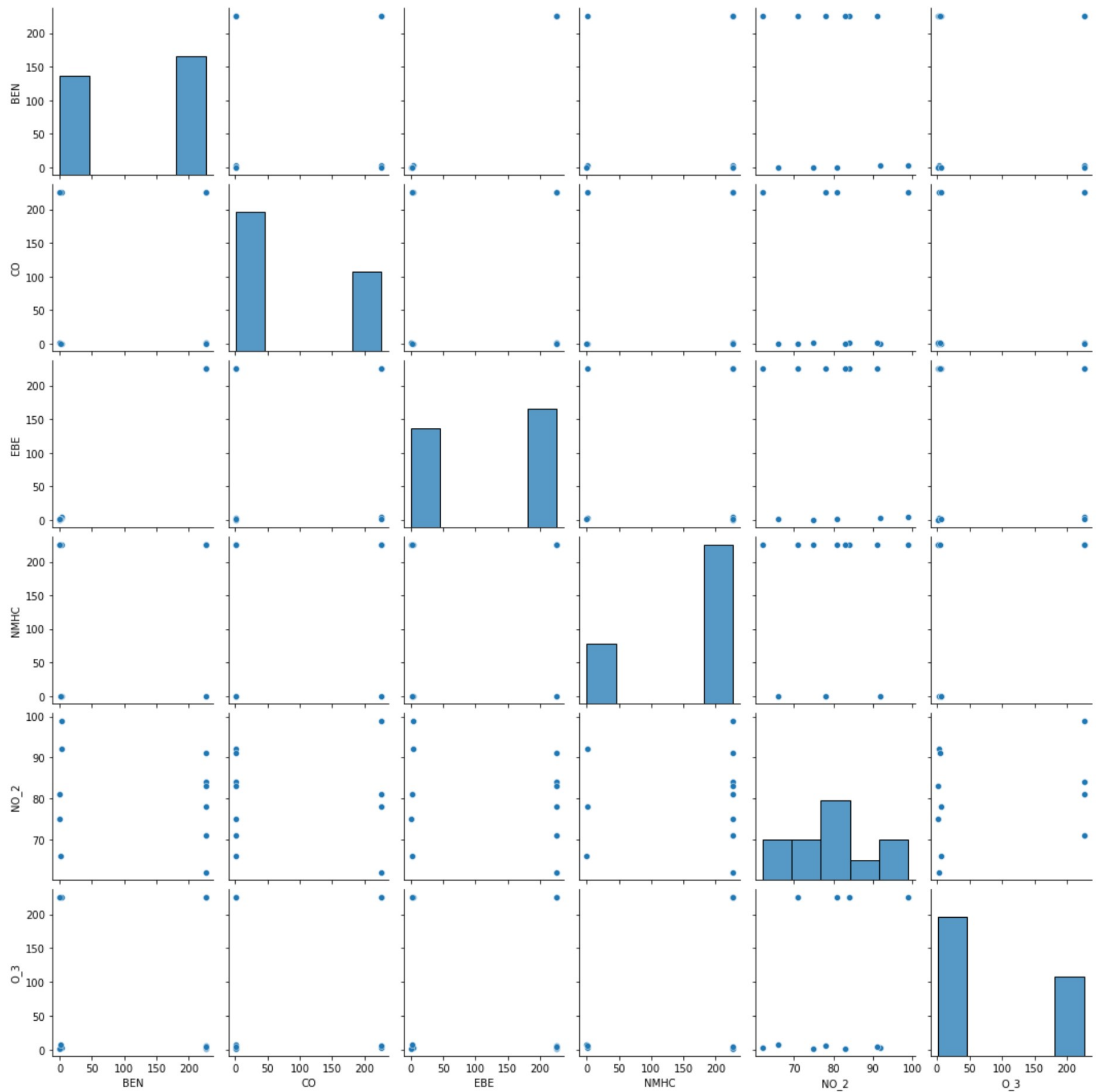
In [17]:

Out[17]: <AxesSubplot:>



In [18]:

Out[18]: <seaborn.axisgrid.PairGrid at 0x24c9910ee50>



In [20]:
```
x=d[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2']]
```

In [21]:
```
from sklearn.model_selection import train_test_split
```

In [22]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

Out[22]: LinearRegression()

In [23]:

1.5631940186722204e-13

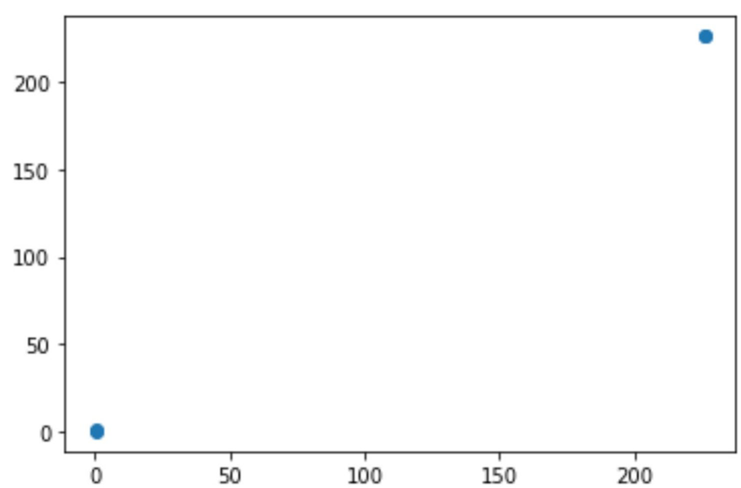
In [24]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[24]:

	Co-efficient
BEN	-2.292249e-17
CO	1.000000e+00
EBE	-1.529799e-16
NMHC	-6.182170e-16
NO_2	-4.434283e-16

In [25]:
```
prediction=lr.predict(x_test)
```

Out[25]: <matplotlib.collections.PathCollection at 0x24c9d1ba580>



In [26]:

1.0

In [27]:

In [28]:
```
rr=Ridge(alpha=10)
```

Out[28]: Ridge(alpha=10)

In [29]:

Out[29]: 0.9999983133518464

In [30]:
```
la=Lasso(alpha=10)
```

Out[30]: Lasso(alpha=10)

In [31]:

Out[31]: 0.9999988983354677

In [32]:
```
a1=b.head(6000)
```

Out[32]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH
0	2011-11-01 01:00:00	226.0	1.0	226.0	226.00	154.0	84.0	226.0	226.0	226.0	6.0	226.00
1	2011-11-01 01:00:00	2.5	0.4	3.5	0.26	68.0	92.0	3.0	40.0	24.0	9.0	1.54
2	2011-11-01 01:00:00	2.9	226.0	3.8	226.00	96.0	99.0	226.0	226.0	226.0	226.0	226.00
3	2011-11-01 01:00:00	226.0	0.6	226.0	226.00	60.0	83.0	2.0	226.0	226.0	226.0	226.00
4	2011-11-01 01:00:00	226.0	226.0	226.0	226.00	44.0	62.0	3.0	226.0	226.0	3.0	226.00
...	...	...	...	...	...	...	...	...	...	...	...	...
5995	2011-11-11 10:00:00	226.0	0.6	226.0	226.00	133.0	81.0	8.0	226.0	226.0	226.0	226.00
5996	2011-11-11 10:00:00	226.0	0.6	226.0	226.00	112.0	86.0	226.0	41.0	226.0	8.0	226.00
5997	2011-11-11 10:00:00	226.0	226.0	226.0	0.39	34.0	21.0	2.0	226.0	226.0	226.0	1.99
5998	2011-11-11 10:00:00	226.0	226.0	226.0	226.00	88.0	59.0	9.0	226.0	226.0	226.0	226.00
5999	2011-11-11 10:00:00	226.0	226.0	226.0	226.00	116.0	81.0	3.0	44.0	226.0	226.0	226.00

6000 rows × 14 columns

In [39]:
```
e=a1[['BEN', 'CO', 'EBE','NMHC', 'NO_2','O_3',
```

In [40]:
```
f=e.iloc[:,0:14]
```

In [41]:

In [42]:
```
logr=LogisticRegression(max_iter=10000)
```

Out[42]: LogisticRegression(max_iter=10000)

In [43]:
```
from sklearn.model_selection import train_test_split
```

In [49]:

In [50]:
```
prediction=logr.predict(i)
```

[28079050]

In [51]:

Out[51]:
```
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 [52]:

Out[52]:
```
0.0
```

In [53]:

Out[53]:
```
0.0
```

In [54]:

Out[54]:
```
0.9827777777777778
```

In [55]:
```python
from sklearn.linear_model import ElasticNet
en=ElasticNet()
```

Out[55]:
```
ElasticNet()
```

In [56]:

```
[-0.18268316  0.99988474  0.18344829  0.          -0.        ]
```

In [57]:

```
-0.17135863671397544
```

In [58]:
```python
prediction=en.predict(x_test)
```

```
0.9999989342451869
```

In [59]:
```python
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

Out[59]:
```
RandomForestClassifier()
```

In [60]:
```python
parameters={'max_depth':[1,2,3,4,5],
 'min_samples_leaf':[5,10,15,20,25],
 'n_estimators':[10,20,30,40,50]
```

In [61]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="acc
```

Out[61]:
```
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 [62]:

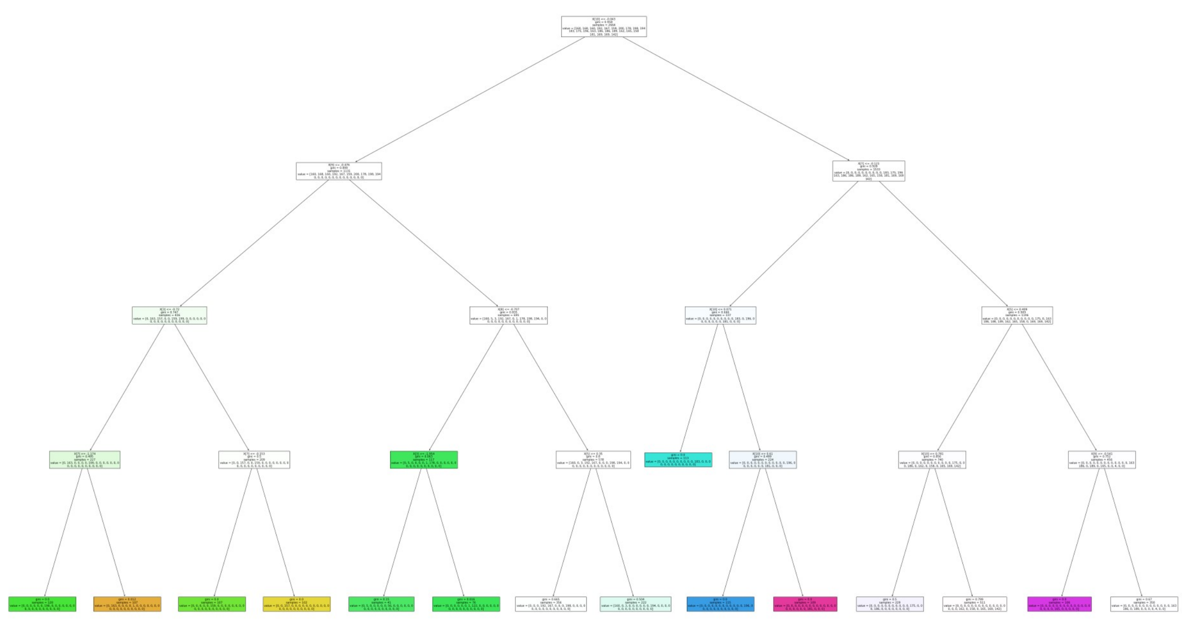
Out[62]:
```
0.9995238095238095
```

In [63]:

In [64]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,50))
```

Out[64]:
```
[Text(2271.8571428571427, 2446.2, 'X[10] <= -0.043\ngini = 0.958\nsamples = 2
664\nvalue = [160, 168, 160, 192, 167, 159, 200, 178, 198, 194\n183, 175, 19
6, 163, 186, 186, 189, 162, 165, 158\n181, 169, 169, 142]'),
 Text(1275.4285714285713, 1902.6, 'X[9] <= -0.476\ngini = 0.899\nsamples = 11
31\nvalue = [160, 168, 160, 192, 167, 159, 200, 178, 198, 194\n0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(637.7142857142857, 1359.0, 'X[3] <= -0.72\ngini = 0.747\nsamples = 436\
nvalue = [0, 163, 157, 0, 0, 159, 199, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(318.85714285714283, 815.3999999999999, 'X[7] <= -1.174\ngini = 0.495\ns
amples = 227\nvalue = [0, 163, 0, 0, 0, 0, 199, 0, 0, 0, 0, 0, 0, 0\n0, 0,
0, 0, 0, 0, 0, 0, 0]'),
 Text(159.42857142857142, 271.7999999999997, 'gini = 0.0\nsamples = 120\nvalu
e = [0, 0, 0, 0, 0, 0, 198, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(478.2857142857142, 271.7999999999997, 'gini = 0.012\nsamples = 107\nval
ue = [0, 163, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(956.5714285714284, 815.3999999999999, 'X[7] <= -0.153\ngini = 0.5\nsamp
les = 209\nvalue = [0, 0, 157, 0, 0, 159, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0,
0, 0, 0, 0, 0, 0]'),
 Text(797.1428571428571, 271.7999999999997, 'gini = 0.0\nsamples = 107\nvalue
= [0, 0, 0, 0, 0, 159, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1116.0, 271.7999999999997, 'gini = 0.0\nsamples = 102\nvalue = [0, 0, 1
57, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1913.1428571428569, 1359.0, 'X[8] <= -0.707\ngini = 0.835\nsamples = 69
5\nvalue = [160, 5, 3, 192, 167, 0, 1, 178, 198, 194, 0, 0\n0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0]'),
 Text(1594.2857142857142, 815.3999999999999, 'X[3] <= -1.954\ngini = 0.063\ns
amples = 117\nvalue = [0, 5, 0, 0, 0, 0, 1, 178, 0, 0, 0, 0, 0, 0\n0, 0,
0, 0, 0, 0, 0, 0, 0]'),
 Text(1434.8571428571427, 271.7999999999997, 'gini = 0.15\nsamples = 41\nvalu
e = [0, 5, 0, 0, 0, 0, 0, 56, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1753.7142857142856, 271.7999999999997, 'gini = 0.016\nsamples = 76\nval
ue = [0, 0, 0, 0, 0, 0, 1, 122, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(2232.0, 815.3999999999999, 'X[5] <= 0.35\ngini = 0.8\nsamples = 578\nva
lue = [160, 0, 3, 192, 167, 0, 0, 0, 198, 194, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(2072.5714285714284, 271.7999999999997, 'gini = 0.665\nsamples = 356\nva
lue = [0, 0, 0, 192, 167, 0, 0, 0, 198, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(2391.428571428571, 271.7999999999997, 'gini = 0.504\nsamples = 222\nval
ue = [160, 0, 3, 0, 0, 0, 0, 0, 0, 194, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(3268.285714285714, 1902.6, 'X[7] <= -0.121\ngini = 0.928\nsamples = 153
3\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 183, 175, 196\n163, 186, 186, 189,
162, 165, 158, 181, 169, 169\n142]'),
 Text(2710.285714285714, 1359.0, 'X[10] <= 0.071\ngini = 0.666\nsamples = 33
7\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 183, 0, 196, 0\n0, 0, 0, 0, 0, 1
81, 0, 0, 0]'),
```

```
 Text(2550.8571428571427, 815.3999999999999, 'gini = 0.0\nsamples = 113\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 183, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(2869.7142857142853, 815.3999999999999, 'X[10] <= 0.61\ngini = 0.499\nsa
mples = 224\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 196, 0\n0, 0, 0, 0,
0, 0, 181, 0, 0, 0]'),
 Text(2710.285714285714, 271.7999999999997, 'gini = 0.0\nsamples = 115\nvalue
= [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 196, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(3029.142857142857, 271.7999999999997, 'gini = 0.0\nsamples = 109\nvalue
= [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 181, 0, 0,
0]'),
 Text(3826.2857142857138, 1359.0, 'X[5] <= 0.409\ngini = 0.909\nsamples = 119
6\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 175, 0, 163\n186, 186, 189, 162,
165, 158, 0, 169, 169, 142]'),
 Text(3507.428571428571, 815.3999999999999, 'X[10] <= 0.781\ngini = 0.856\nsa
mples = 740\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 175, 0, 0\n0, 186, 0,
162, 0, 158, 0, 165, 169, 142]'),
 Text(3347.9999999999995, 271.7999999999997, 'gini = 0.5\nsamples = 229\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 175, 0, 0\n0, 186, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(3666.8571428571427, 271.7999999999997, 'gini = 0.799\nsamples = 511\nva
lue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 162, 0, 158, 0, 16
5, 169, 142]'),
 Text(4145.142857142857, 815.3999999999999, 'X[9] <= -0.541\ngini = 0.752\nsa
mples = 456\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 163\n186, 0, 18
9, 0, 165, 0, 0, 4, 0, 0]'),
 Text(3985.7142857142853, 271.7999999999997, 'gini = 0.0\nsamples = 106\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 165, 0, 0, 0, 0,
0]'),
 Text(4304.571428571428, 271.7999999999997, 'gini = 0.67\nsamples = 350\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 163\n186, 0, 189, 0, 0, 0, 0, 4,
0, 0]')]
```



From this observation I had observe that the LASSO is a highest accuracy of 0.9999988983354677

In [ ]: