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	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_2
0	2017-06-01 01:00:00	NaN	NaN	0.3	NaN	NaN	4.0	38.0	NaN	NaN	NaN	NaN	5.0
1	2017-06-01 01:00:00	0.6	NaN	0.3	0.4	0.08	3.0	39.0	NaN	71.0	22.0	9.0	7.0
2	2017-06-01 01:00:00	0.2	NaN	NaN	0.1	NaN	1.0	14.0	NaN	NaN	NaN	NaN	NaN
3	2017-06-01 01:00:00	NaN	NaN	0.2	NaN	NaN	1.0	9.0	NaN	91.0	NaN	NaN	NaN
4	2017-06-01 01:00:00	NaN	NaN	NaN	NaN	NaN	1.0	19.0	NaN	69.0	NaN	NaN	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
210115	2017-08-01 00:00:00	NaN	NaN	0.2	NaN	NaN	1.0	27.0	NaN	65.0	NaN	NaN	NaN
210116	2017-08-01 00:00:00	NaN	NaN	0.2	NaN	NaN	1.0	14.0	NaN	NaN	73.0	NaN	7.0
210117	2017-08-01 00:00:00	NaN	NaN	NaN	NaN	NaN	1.0	4.0	NaN	83.0	NaN	NaN	NaN
210118	2017-08-01 00:00:00	NaN	NaN	NaN	NaN	NaN	1.0	11.0	NaN	78.0	NaN	NaN	NaN
210119	2017-08-01 00:00:00	NaN	NaN	NaN	NaN	NaN	1.0	14.0	NaN	77.0	60.0	NaN	NaN

210120 rows × 16 columns

In [3]:
```
.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 210120 entries, 0 to 210119
Data columns (total 16 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     210120 non-null  object 
 1   BEN      50201 non-null   float64
 2   CH4      6410 non-null    float64
 3   CO       87001 non-null   float64
 4   EBE      49973 non-null   float64
 5   NMHC     25472 non-null   float64
 6   NO       209065 non-null  float64
 7   NO_2     209065 non-null  float64
 8   NOx      52818 non-null   float64
 9   O_3      121398 non-null  float64
 10  PM10     104141 non-null  float64
 11  PM25     52023 non-null   float64
 12  SO_2     86803 non-null   float64
 13  TCH      25472 non-null   float64
 14  TOL      50117 non-null   float64
 15  station  210120 non-null  int64  
dtypes: float64(14), int64(1), object(1)
memory usage: 25.6+ MB
```

In [4]:
```
b=a.fillna(value=172)
```

Out[4]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	S
0	2017-06-01 01:00:00	172.0	172.0	0.3	172.0	172.00	4.0	38.0	172.0	172.0	172.0	172.0	
1	2017-06-01 01:00:00	0.6	172.0	0.3	0.4	0.08	3.0	39.0	172.0	71.0	22.0	9.0	
2	2017-06-01 01:00:00	0.2	172.0	172.0	0.1	172.00	1.0	14.0	172.0	172.0	172.0	172.0	1
3	2017-06-01 01:00:00	172.0	172.0	0.2	172.0	172.00	1.0	9.0	172.0	91.0	172.0	172.0	1
4	2017-06-01 01:00:00	172.0	172.0	172.0	172.0	172.00	1.0	19.0	172.0	69.0	172.0	172.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
210115	2017-08-01 00:00:00	172.0	172.0	0.2	172.0	172.00	1.0	27.0	172.0	65.0	172.0	172.0	1
210116	2017-08-01 00:00:00	172.0	172.0	0.2	172.0	172.00	1.0	14.0	172.0	172.0	73.0	172.0	
210117	2017-08-01 00:00:00	172.0	172.0	172.0	172.0	172.00	1.0	4.0	172.0	83.0	172.0	172.0	1
210118	2017-08-01 00:00:00	172.0	172.0	172.0	172.0	172.00	1.0	11.0	172.0	78.0	172.0	172.0	1
210119	2017-08-01 00:00:00	172.0	172.0	172.0	172.0	172.00	1.0	14.0	172.0	77.0	60.0	172.0	1

210120 rows × 16 columns

In [5]:
```

```

Out[5]:
```
Index(['date', 'BEN', 'CH4', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'NOx', 'O_3',
       'PM10', 'PM25', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

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

Out[6]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_2
0	2017-06-01 01:00:00	172.0	172.0	0.3	172.0	172.00	4.0	38.0	172.0	172.0	172.0	172.0	5.0
1	2017-06-01 01:00:00	0.6	172.0	0.3	0.4	0.08	3.0	39.0	172.0	71.0	22.0	9.0	7.0
2	2017-06-01 01:00:00	0.2	172.0	172.0	0.1	172.00	1.0	14.0	172.0	172.0	172.0	172.0	172.0
3	2017-06-01 01:00:00	172.0	172.0	0.2	172.0	172.00	1.0	9.0	172.0	91.0	172.0	172.0	172.0
4	2017-06-01 01:00:00	172.0	172.0	172.0	172.0	172.00	1.0	19.0	172.0	69.0	172.0	172.0	2.0
5	2017-06-01 01:00:00	0.1	172.0	0.3	0.2	172.00	1.0	26.0	172.0	70.0	26.0	172.0	1.0
6	2017-06-01 01:00:00	0.3	172.0	0.2	0.1	0.17	1.0	19.0	172.0	79.0	23.0	9.0	3.0
7	2017-06-01 01:00:00	172.0	172.0	172.0	172.0	172.00	1.0	9.0	172.0	87.0	172.0	172.0	172.0
8	2017-06-01 01:00:00	172.0	172.0	0.3	172.0	172.00	3.0	30.0	172.0	70.0	172.0	172.0	172.0
9	2017-06-01 01:00:00	172.0	172.0	0.1	172.0	172.00	1.0	15.0	172.0	172.0	22.0	172.0	10.0
10	2017-06-01 01:00:00	0.7	172.0	172.0	1.0	172.00	1.0	25.0	172.0	172.0	21.0	10.0	2.0

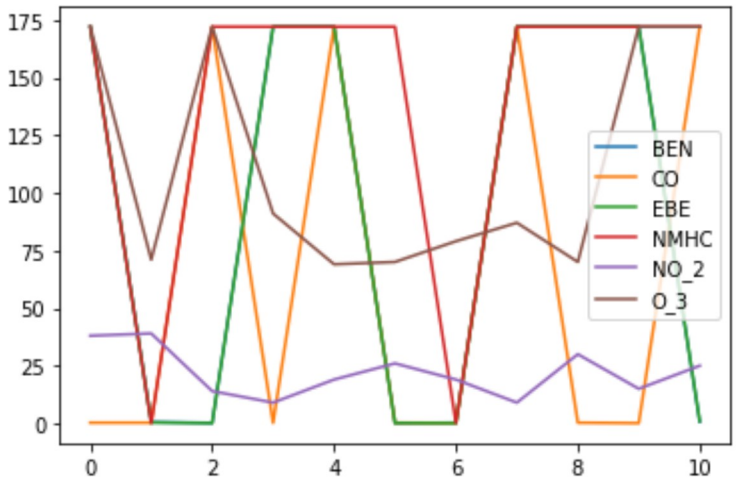
In [7]:
```
d=c[['BEN','CO','EBE','NMHC','NO_2','O_3']]
```

Out[7]:

	BEN	CO	EBE	NMHC	NO_2	O_3
0	172.0	0.3	172.0	172.00	38.0	172.0
1	0.6	0.3	0.4	0.08	39.0	71.0
2	0.2	172.0	0.1	172.00	14.0	172.0
3	172.0	0.2	172.0	172.00	9.0	91.0
4	172.0	172.0	172.0	172.00	19.0	69.0
5	0.1	0.3	0.2	172.00	26.0	70.0
6	0.3	0.2	0.1	0.17	19.0	79.0
7	172.0	172.0	172.0	172.00	9.0	87.0
8	172.0	0.3	172.0	172.00	30.0	70.0
9	172.0	0.1	172.0	172.00	15.0	172.0
10	0.7	172.0	1.0	172.00	25.0	172.0

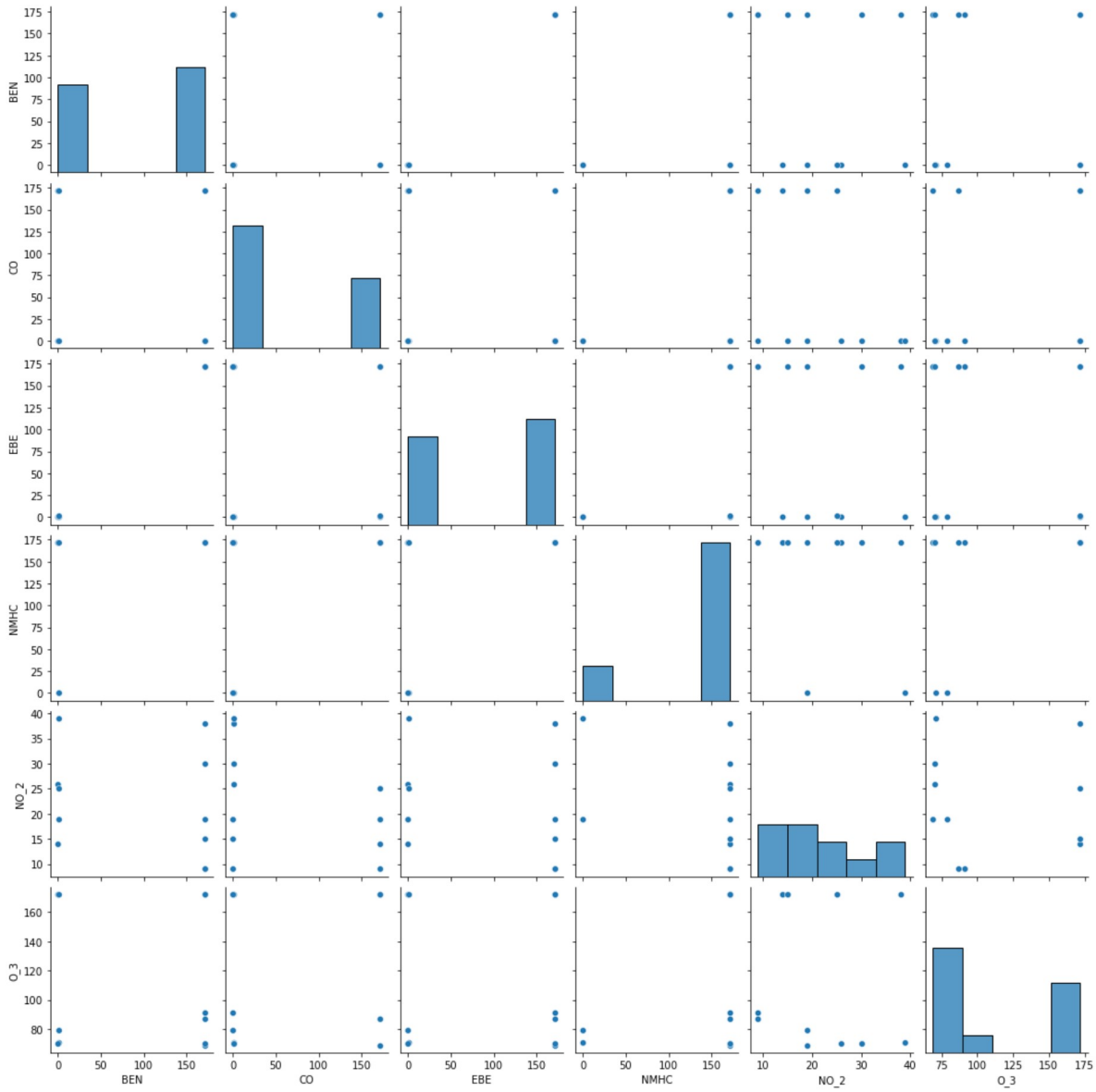
In [8]:

Out[8]: <AxesSubplot:>



In [9]:

Out[9]: <seaborn.axisgrid.PairGrid at 0x16c0c19b4c0>



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]:

```
7.027267656667391e-12
```

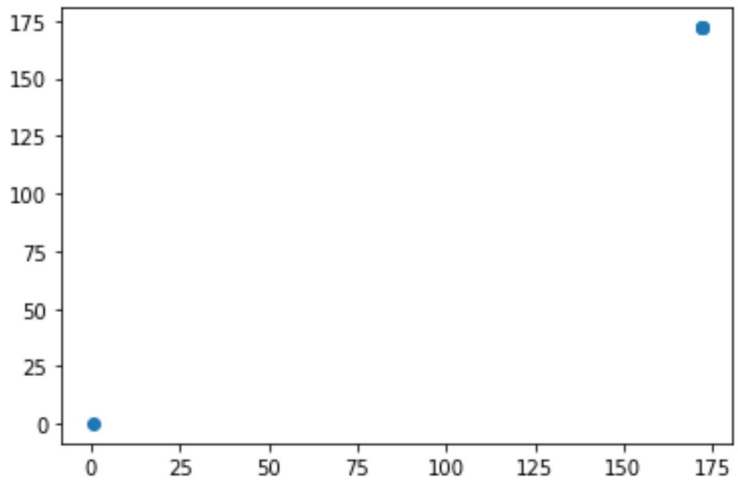
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	-3.512958e-11
CO	1.000000e+00
EBE	3.514969e-11
NMHC	-6.107000e-14
NO_2	5.495238e-16

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

Out[15]: <matplotlib.collections.PathCollection at 0x16c0ed10e20>



In [16]:

```
1.0
```

In [17]:

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

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

In [19]:

Out[19]:
```
0.9999986435877276
```

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

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

In [21]:

Out[21]:
```
0.9999772585745981
```

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

Out[22]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_
0	2017-06-01 01:00:00	172.0	172.0	0.3	172.0	172.00	4.0	38.0	172.0	172.0	172.0	172.0	5
1	2017-06-01 01:00:00	0.6	172.0	0.3	0.4	0.08	3.0	39.0	172.0	71.0	22.0	9.0	7
2	2017-06-01 01:00:00	0.2	172.0	172.0	0.1	172.00	1.0	14.0	172.0	172.0	172.0	172.0	172
3	2017-06-01 01:00:00	172.0	172.0	0.2	172.0	172.00	1.0	9.0	172.0	91.0	172.0	172.0	172
4	2017-06-01 01:00:00	172.0	172.0	172.0	172.0	172.00	1.0	19.0	172.0	69.0	172.0	172.0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	
6995	2017-06-13 06:00:00	172.0	172.0	0.2	172.0	172.00	1.0	9.0	172.0	84.0	172.0	172.0	172
6996	2017-06-13 06:00:00	172.0	172.0	172.0	172.0	172.00	1.0	13.0	172.0	172.0	7.0	172.0	9
6997	2017-06-13 06:00:00	172.0	172.0	172.0	172.0	172.00	1.0	11.0	172.0	172.0	20.0	17.0	172
6998	2017-06-13 06:00:00	172.0	172.0	172.0	172.0	172.00	1.0	2.0	172.0	172.0	8.0	4.0	172
6999	2017-06-13 06:00:00	172.0	172.0	172.0	172.0	172.00	1.0	3.0	172.0	76.0	172.0	172.0	172

7000 rows × 16 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]:
```
0.9295238095238095
```

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

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_coordinate_d
escent.py:530: ConvergenceWarning: Objective did not converge. You might want
to increase the number of iterations. Duality gap: 2.8279111983000207, tolera
nce: 2.5288994283597512
  model = cd_fast.enet_coordinate_descent(
```

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

In [35]:

```
[-0.08805575  0.99952198  0.08786105  0.         -0.        ]
```

In [36]:

```
0.02624026845406391
```

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

```
0.9999990835844101
```

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]:

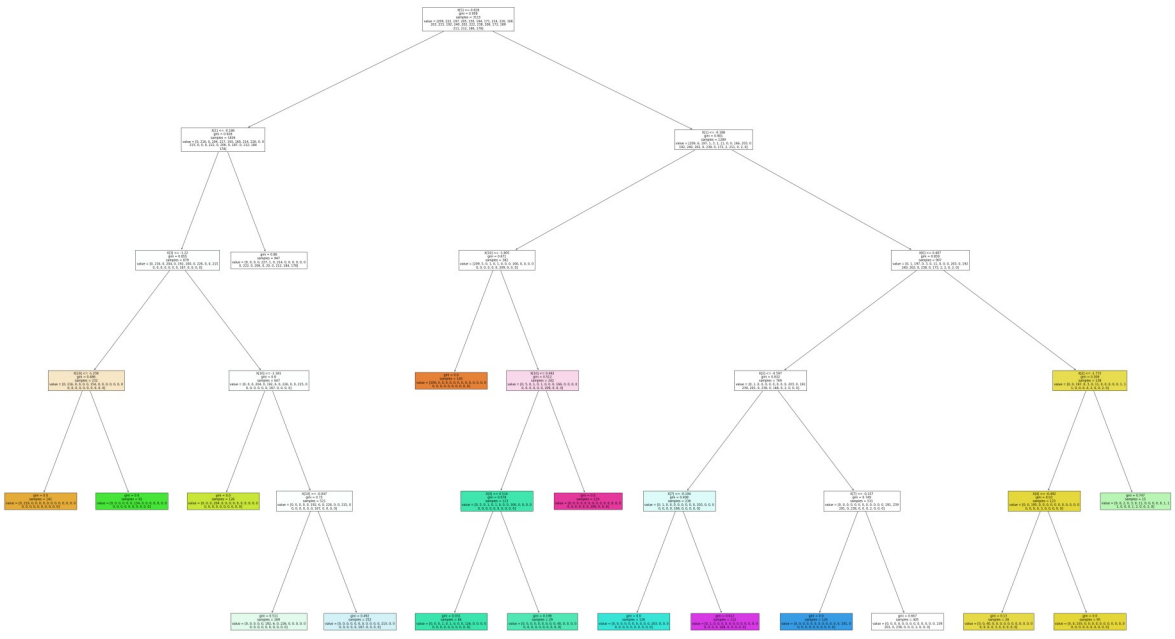
Out[41]:
```
0.9914285714285714
```

In [42]:

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

Out[43]:
```
[Text(1781.3076923076922, 2491.5, 'X[5] <= 0.828\ngini = 0.958\nsamples = 311
5\nvalue = [209, 222, 197, 205, 230, 194, 171, 214, 226, 166\n203, 215, 192,
240, 202, 222, 238, 206, 172, 189\n211, 212, 186, 178]'),
 Text(858.4615384615383, 2038.5, 'X[1] <= -0.186\ngini = 0.928\nsamples = 182
6\nvalue = [0, 216, 0, 204, 227, 193, 160, 214, 226, 0, 0\n215, 0, 0, 0, 222,
0, 206, 0, 187, 0, 212, 184\n178]'),
 Text(686.7692307692307, 1585.5, 'X[3] <= -1.22\ngini = 0.855\nsamples = 879\
nvalue = [0, 216, 0, 204, 0, 192, 160, 0, 226, 0, 0, 215\n0, 0, 0, 0, 0, 0,
0, 167, 0, 0, 0, 0]'),
 Text(343.38461538461536, 1132.5, 'X[10] <= -1.238\ngini = 0.486\nsamples = 2
32\nvalue = [0, 216, 0, 0, 0, 0, 154, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(171.69230769230768, 679.5, 'gini = 0.0\nsamples = 141\nvalue = [0, 216,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(515.0769230769231, 679.5, 'gini = 0.0\nsamples = 91\nvalue = [0, 0, 0,
0, 0, 0, 154, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1030.1538461538462, 1132.5, 'X[10] <= -1.181\ngini = 0.8\nsamples = 64
7\nvalue = [0, 0, 0, 204, 0, 192, 6, 0, 226, 0, 0, 215, 0\n0, 0, 0, 0, 0, 0,
167, 0, 0, 0, 0]'),
 Text(858.4615384615383, 679.5, 'gini = 0.0\nsamples = 126\nvalue = [0, 0, 0,
204, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1201.8461538461538, 679.5, 'X[10] <= -0.047\ngini = 0.75\nsamples = 52
1\nvalue = [0, 0, 0, 0, 0, 192, 6, 0, 226, 0, 0, 215, 0\n0, 0, 0, 0, 0, 0, 16
7, 0, 0, 0, 0]'),
 Text(1030.1538461538462, 226.5, 'gini = 0.511\nsamples = 269\nvalue = [0, 0,
0, 0, 0, 192, 6, 0, 226, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1373.5384615384614, 226.5, 'gini = 0.492\nsamples = 252\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 215, 0, 0\n0, 0, 0, 0, 0, 167, 0, 0, 0, 0]'),
 Text(1030.1538461538462, 1585.5, 'gini = 0.86\nsamples = 947\nvalue = [0, 0,
0, 0, 227, 1, 0, 214, 0, 0, 0, 0, 0, 0\n0, 222, 0, 206, 0, 20, 0, 212, 184, 1
78]'),
 Text(2704.1538461538457, 2038.5, 'X[1] <= -0.188\ngini = 0.901\nsamples = 12
89\nvalue = [209, 6, 197, 1, 3, 1, 11, 0, 0, 166, 203, 0\n192, 240, 202, 0, 2
38, 0, 172, 2, 211, 0, 2, 0]'),
 Text(1888.6153846153845, 1585.5, 'X[10] <= -1.805\ngini = 0.671\nsamples = 3
82\nvalue = [209, 5, 0, 1, 0, 1, 0, 0, 0, 166, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
209, 0, 0, 0]'),
 Text(1716.9230769230767, 1132.5, 'gini = 0.0\nsamples = 140\nvalue = [209,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2060.3076923076924, 1132.5, 'X[10] <= 0.492\ngini = 0.512\nsamples = 24
2\nvalue = [0, 5, 0, 1, 0, 1, 0, 0, 0, 166, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 20
9, 0, 0, 0]'),
 Text(1888.6153846153845, 679.5, 'X[4] <= 0.514\ngini = 0.078\nsamples = 113\
nvalue = [0, 5, 0, 1, 0, 1, 0, 0, 0, 166, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(1716.9230769230767, 226.5, 'gini = 0.031\nsamples = 84\nvalue = [0, 0,
0, 1, 0, 1, 0, 0, 0, 126, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2060.3076923076924, 226.5, 'gini = 0.198\nsamples = 29\nvalue = [0, 5,
0, 0, 0, 0, 0, 0, 0, 40, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2232.0, 679.5, 'gini = 0.0\nsamples = 129\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 209, 0, 0, 0]'),
 Text(3519.6923076923076, 1585.5, 'X[6] <= 0.407\ngini = 0.859\nsamples = 90
7\nvalue = [0, 1, 197, 0, 3, 0, 11, 0, 0, 0, 203, 0, 192\n240, 202, 0, 238,
```

```
0, 172, 2, 2, 0, 2, 0]'),
 Text(2918.7692307692305, 1132.5, 'X[2] <= -0.597\ngini = 0.832\nsamples = 76
9\nvalue = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 203, 0, 191\n239, 201, 0, 238, 0, 1
68, 0, 2, 0, 0, 0]'),
 Text(2575.3846153846152, 679.5, 'X[7] <= -0.194\ngini = 0.498\nsamples = 23
8\nvalue = [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 203, 0, 0, 0\n0, 0, 0, 0, 168, 0,
0, 0, 0, 0]'),
 Text(2403.6923076923076, 226.5, 'gini = 0.0\nsamples = 126\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 203, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2747.076923076923, 226.5, 'gini = 0.012\nsamples = 112\nvalue = [0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 168, 0, 0, 0, 0, 0]'),
 Text(3262.1538461538457, 679.5, 'X[7] <= -0.157\ngini = 0.749\nsamples = 53
1\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 191, 239\n201, 0, 238, 0, 0,
0, 2, 0, 0, 0]'),
 Text(3090.461538461538, 226.5, 'gini = 0.0\nsamples = 126\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 191, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(3433.8461538461534, 226.5, 'gini = 0.667\nsamples = 405\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 239\n201, 0, 238, 0, 0, 0, 2, 0, 0, 0]'),
 Text(4120.615384615385, 1132.5, 'X[2] <= -1.772\ngini = 0.209\nsamples = 13
8\nvalue = [0, 0, 197, 0, 3, 0, 11, 0, 0, 0, 0, 0, 1, 1\n1, 0, 0, 0, 4, 2, 0,
0, 2, 0]'),
 Text(3948.9230769230767, 679.5, 'X[4] <= -0.492\ngini = 0.03\nsamples = 123\
nvalue = [0, 0, 195, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 3, 0, 0, 0,
0, 0]'),
 Text(3777.230769230769, 226.5, 'gini = 0.13\nsamples = 28\nvalue = [0, 0, 4
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 3, 0, 0, 0, 0, 0]'),
 Text(4120.615384615385, 226.5, 'gini = 0.0\nsamples = 95\nvalue = [0, 0, 15
5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(4292.307692307692, 679.5, 'gini = 0.747\nsamples = 15\nvalue = [0, 0,
2, 0, 3, 0, 11, 0, 0, 0, 0, 0, 1, 1\n1, 0, 0, 0, 1, 2, 0, 0, 2, 0]')]
```



From this observation I had observe that the

ELASTICNET is a highest accuracy of 0.9999990835844101

In [ ]: