In [130]:

```
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 [706]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2017.csv")
a
```

Out[706]:

	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 [707]:

```
a.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 [708]:

```
b=a.fillna(value=104)
b
```

Out[708]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	S
0	2017-06-01 01:00:00	104.0	104.0	0.3	104.0	104.00	4.0	38.0	104.0	104.0	104.0	104.0	
1	2017-06-01 01:00:00	0.6	104.0	0.3	0.4	0.08	3.0	39.0	104.0	71.0	22.0	9.0	
2	2017-06-01 01:00:00	0.2	104.0	104.0	0.1	104.00	1.0	14.0	104.0	104.0	104.0	104.0	1
3	2017-06-01 01:00:00	104.0	104.0	0.2	104.0	104.00	1.0	9.0	104.0	91.0	104.0	104.0	1
4	2017-06-01 01:00:00	104.0	104.0	104.0	104.0	104.00	1.0	19.0	104.0	69.0	104.0	104.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
210115	2017-08-01 00:00:00	104.0	104.0	0.2	104.0	104.00	1.0	27.0	104.0	65.0	104.0	104.0	1
210116	2017-08-01 00:00:00	104.0	104.0	0.2	104.0	104.00	1.0	14.0	104.0	104.0	73.0	104.0	
210117	2017-08-01 00:00:00	104.0	104.0	104.0	104.0	104.00	1.0	4.0	104.0	83.0	104.0	104.0	1
210118	2017-08-01 00:00:00	104.0	104.0	104.0	104.0	104.00	1.0	11.0	104.0	78.0	104.0	104.0	1
210119	2017-08-01 00:00:00	104.0	104.0	104.0	104.0	104.00	1.0	14.0	104.0	77.0	60.0	104.0	1

210120 rows × 16 columns

In [709]:

```
b.columns
```

Out[709]:

```
Index(['date', 'BEN', 'CH4', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'NOx', 'O_
3',
       'PM10', 'PM25', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [710]:

```
c=b.head(10)
c
```

Out[710]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO_2
0	2017-06-01 01:00:00	104.0	104.0	0.3	104.0	104.00	4.0	38.0	104.0	104.0	104.0	104.0	5.0
1	2017-06-01 01:00:00	0.6	104.0	0.3	0.4	0.08	3.0	39.0	104.0	71.0	22.0	9.0	7.0
2	2017-06-01 01:00:00	0.2	104.0	104.0	0.1	104.00	1.0	14.0	104.0	104.0	104.0	104.0	104.0
3	2017-06-01 01:00:00	104.0	104.0	0.2	104.0	104.00	1.0	9.0	104.0	91.0	104.0	104.0	104.0
4	2017-06-01 01:00:00	104.0	104.0	104.0	104.0	104.00	1.0	19.0	104.0	69.0	104.0	104.0	2.0
5	2017-06-01 01:00:00	0.1	104.0	0.3	0.2	104.00	1.0	26.0	104.0	70.0	26.0	104.0	1.0
6	2017-06-01 01:00:00	0.3	104.0	0.2	0.1	0.17	1.0	19.0	104.0	79.0	23.0	9.0	3.0
7	2017-06-01 01:00:00	104.0	104.0	104.0	104.0	104.00	1.0	9.0	104.0	87.0	104.0	104.0	104.0
8	2017-06-01 01:00:00	104.0	104.0	0.3	104.0	104.00	3.0	30.0	104.0	70.0	104.0	104.0	104.0
9	2017-06-01 01:00:00	104.0	104.0	0.1	104.0	104.00	1.0	15.0	104.0	104.0	22.0	104.0	10.0

In [711]:

```
d=c[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
d
```

Out[711]:

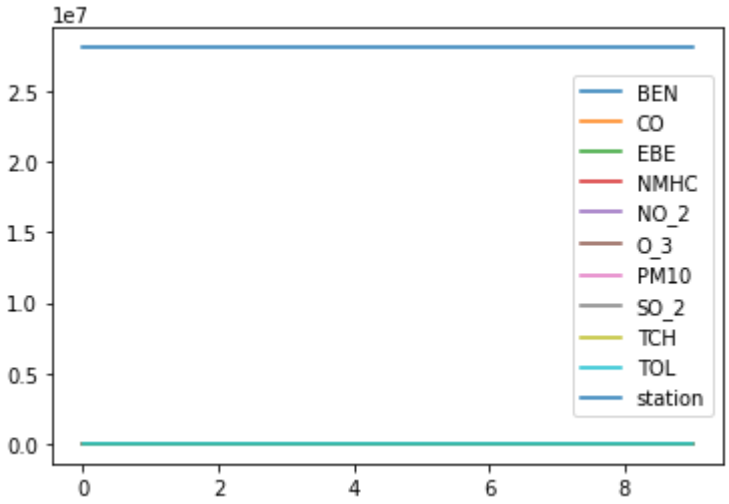
	BEN	CO	EBE	NMHC	NO_2	O_3	PM10	SO_2	TCH	TOL	station
0	104.0	0.3	104.0	104.00	38.0	104.0	104.0	5.0	104.00	104.0	28079004
1	0.6	0.3	0.4	0.08	39.0	71.0	22.0	7.0	1.40	2.9	28079008
2	0.2	104.0	0.1	104.00	14.0	104.0	104.0	104.0	104.00	0.9	28079011
3	104.0	0.2	104.0	104.00	9.0	91.0	104.0	104.0	104.00	104.0	28079016
4	104.0	104.0	104.0	104.00	19.0	69.0	104.0	2.0	104.00	104.0	28079017
5	0.1	0.3	0.2	104.00	26.0	70.0	26.0	1.0	104.00	0.3	28079018
6	0.3	0.2	0.1	0.17	19.0	79.0	23.0	3.0	0.86	1.8	28079024
7	104.0	104.0	104.0	104.00	9.0	87.0	104.0	104.0	104.00	104.0	28079027
8	104.0	0.3	104.0	104.00	30.0	70.0	104.0	104.0	104.00	104.0	28079035
9	104.0	0.1	104.0	104.00	15.0	104.0	22.0	10.0	104.00	104.0	28079036

In [712]:

```
d.plot.line()
```

Out[712]:

```
<AxesSubplot:>
```

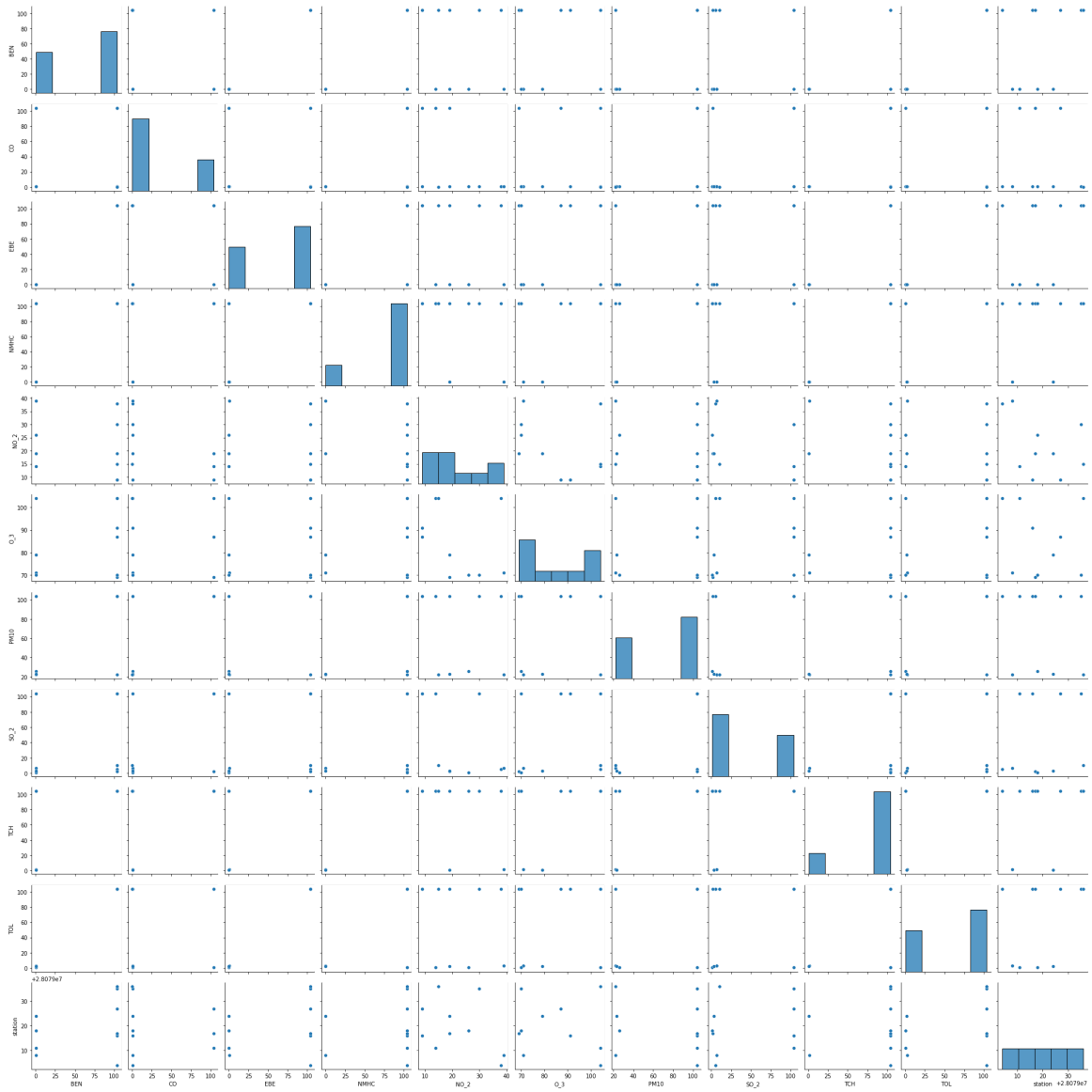


In [713]:

```
sns.pairplot(d)
```

Out[713]:

```
<seaborn.axisgrid.PairGrid at 0x1189b997220>
```



In [714]:

```
x=d[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2']]
y=d['TCH']
```

In [715]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [716]:

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[716]:

```
LinearRegression()
```

In [717]:

```
print(lr.intercept_)
```

```
0.6911297438408894
```

In [718]:

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

Out[718]:

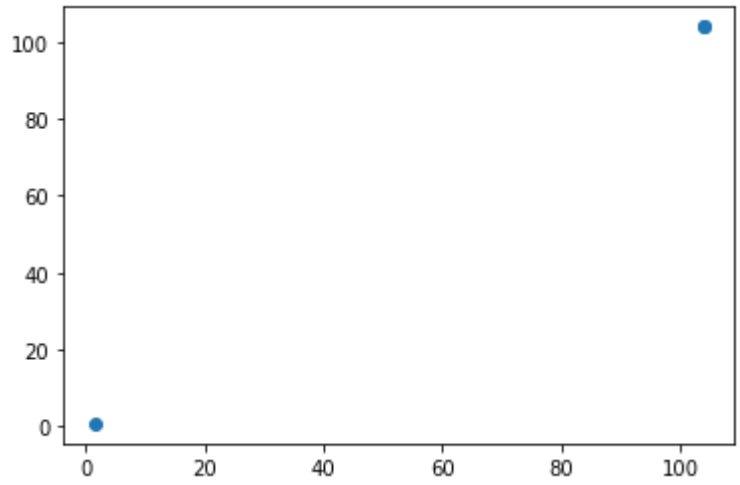
	Co-efficient
BEN	-1.546672e-14
CO	-1.670605e-16
EBE	1.526611e-14
NMHC	9.933545e-01
NO_2	1.543467e-16

In [719]:

```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[719]:

```
<matplotlib.collections.PathCollection at 0x118ae4d4f40>
```



In [720]:

```
print(lr.score(x_test,y_test))
```

```
0.9999435514749918
```

In [721]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [722]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[722]:

```
Ridge(alpha=10)
```

In [723]:

```
rr.score(x_test,y_test)
```

Out[723]:

```
0.9999632503223598
```

In [724]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[724]:

```
Lasso(alpha=10)
```

In [725]:

```
la.score(x_test,y_test)
```

Out[725]:

```
0.9999961074254398
```

In [726]:

```
a1=b.head(7000)
a1
```

Out[726]:

	date	BEN	CH4	CO	EBE	NMHC	NO	NO_2	NOx	O_3	PM10	PM25	SO
0	2017-06-01 01:00:00	104.0	104.0	0.3	104.0	104.00	4.0	38.0	104.0	104.0	104.0	104.0	5
1	2017-06-01 01:00:00	0.6	104.0	0.3	0.4	0.08	3.0	39.0	104.0	71.0	22.0	9.0	7
2	2017-06-01 01:00:00	0.2	104.0	104.0	0.1	104.00	1.0	14.0	104.0	104.0	104.0	104.0	104
3	2017-06-01 01:00:00	104.0	104.0	0.2	104.0	104.00	1.0	9.0	104.0	91.0	104.0	104.0	104
4	2017-06-01 01:00:00	104.0	104.0	104.0	104.0	104.00	1.0	19.0	104.0	69.0	104.0	104.0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	
6995	2017-06-13 06:00:00	104.0	104.0	0.2	104.0	104.00	1.0	9.0	104.0	84.0	104.0	104.0	104
6996	2017-06-13 06:00:00	104.0	104.0	104.0	104.0	104.00	1.0	13.0	104.0	104.0	7.0	104.0	9
6997	2017-06-13 06:00:00	104.0	104.0	104.0	104.0	104.00	1.0	11.0	104.0	104.0	20.0	17.0	104
6998	2017-06-13 06:00:00	104.0	104.0	104.0	104.0	104.00	1.0	2.0	104.0	104.0	8.0	4.0	104
6999	2017-06-13 06:00:00	104.0	104.0	104.0	104.0	104.00	1.0	3.0	104.0	76.0	104.0	104.0	104

7000 rows × 16 columns

In [727]:

```
e=a1[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
```

In [728]:

```
f=e.iloc[:,0:14]
g=e.iloc[:,-1]
```

In [729]:

```
h=StandardScaler().fit_transform(f)
```

In [730]:

```
logr=LogisticRegression(max_iter=10000)
logr.fit(h,g)
```

Out[730]:

```
LogisticRegression(max_iter=10000)
```

In [731]:

```
from sklearn.model_selection import train_test_split
h_train,h_test,g_train,g_test=train_test_split(h,g,test_size=0.3)
```

In [732]:

```
i=[[10,20,30,40,50,60,11,22,33,44,55]]
```

In [733]:

```
prediction=logr.predict(i)
print(prediction)
```

```
[28079059]
```

In [734]:

```
logr.classes_
```

Out[734]:

```
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 [735]:

```
logr.predict_proba(i)[0][0]
```

Out[735]:

```
0.0
```

In [736]:

```
logr.predict_proba(i)[0][1]
```

Out[736]:

```
0.0
```

In [737]:

```
logr.score(h_test,g_test)
```

Out[737]:

```
0.9228571428571428
```

In [738]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[738]:

```
ElasticNet()
```

In [739]:

```
print(en.coef_)
```

```
[ 0.00000000e+00  0.00000000e+00  9.71254138e-05  9.92535011e-01
 -0.00000000e+00]
```

In [740]:

```
print(en.intercept_)
```

```
0.7584256058490553
```

In [741]:

```
prediction=en.predict(x_test)
print(en.score(x_test,y_test))
```

```
0.9999549553875254
```

In [742]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(h_train,g_train)
```

Out[742]:

```
RandomForestClassifier()
```

In [743]:

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

In [744]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(h_train,g_train)
```

Out[744]:

```
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 [745]:

```
grid_search.best_score_
```

Out[745]:

```
0.9840816326530613
```

In [746]:

```
rfc_best=grid_search.best_estimator_
```

In [747]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,50))
plot_tree(rfc_best.estimators_[2],filled=True)
```

Out[747]:

```
[Text(2142.7200000000003, 2491.5, 'X[9] <= -0.196\ngini = 0.958\nsamples =
3105\nvalue = [218, 210, 199, 188, 209, 197, 196, 238, 195, 180\n212, 219,
191, 225, 210, 198, 198, 206, 199, 182\n195, 214, 211, 210]'),
 Text(1160.64, 2038.5, 'X[7] <= -1.197\ngini = 0.833\nsamples = 737\nvalue
= [0, 210, 199, 0, 0, 186, 173, 0, 0, 0, 212, 0\n0, 0, 0, 0, 0, 0, 190, 0,
0, 0, 0, 0]'),
 Text(714.24, 1585.5, 'X[7] <= -1.269\ngini = 0.664\nsamples = 356\nvalue
= [0, 0, 0, 0, 0, 185, 173, 0, 0, 0, 211, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(357.12, 1132.5, 'X[10] <= -0.557\ngini = 0.474\nsamples = 183\nvalue
= [0, 0, 0, 0, 0, 181, 0, 0, 0, 0, 114, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(178.56, 679.5, 'gini = 0.0\nsamples = 111\nvalue = [0, 0, 0, 0, 0, 1
81, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(535.6800000000001, 679.5, 'gini = 0.0\nsamples = 72\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 114, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1071.3600000000001, 1132.5, 'X[10] <= -0.387\ngini = 0.476\nsamples
= 173\nvalue = [0, 0, 0, 0, 0, 4, 173, 0, 0, 0, 97, 0, 0, 0\n0, 0, 0, 0,
0, 0, 0, 0, 0, 0]'),
 Text(892.8, 679.5, 'X[0] <= -1.788\ngini = 0.044\nsamples = 109\nvalue =
[0, 0, 0, 0, 0, 4, 173, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(714.24, 226.5, 'gini = 0.0\nsamples = 93\nvalue = [0, 0, 0, 0, 0, 0,
157, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1071.3600000000001, 226.5, 'gini = 0.32\nsamples = 16\nvalue = [0,
0, 0, 0, 0, 4, 16, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1249.92, 679.5, 'gini = 0.0\nsamples = 64\nvalue = [0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 97, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1607.04, 1585.5, 'X[7] <= -0.134\ngini = 0.668\nsamples = 381\nvalue
= [0, 210, 199, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 190, 0, 0, 0,
0, 0]'),
 Text(1428.48, 1132.5, 'gini = 0.0\nsamples = 135\nvalue = [0, 210, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1785.6, 1132.5, 'X[10] <= -1.323\ngini = 0.505\nsamples = 246\nvalue
= [0, 0, 199, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 190, 0, 0, 0,
0, 0]'),
 Text(1607.04, 679.5, 'gini = 0.0\nsamples = 127\nvalue = [0, 0, 199, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1964.16, 679.5, 'X[2] <= -1.789\ngini = 0.021\nsamples = 119\nvalue
= [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 190, 0, 0, 0, 0,
0]'),
 Text(1785.6, 226.5, 'gini = 0.156\nsamples = 18\nvalue = [0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 22, 0, 0, 0, 0, 0]'),
 Text(2142.7200000000003, 226.5, 'gini = 0.0\nsamples = 101\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 168, 0, 0, 0, 0, 0]'),
 Text(3124.8, 2038.5, 'X[10] <= -1.578\ngini = 0.945\nsamples = 2368\nvalu
e=[[218, 0, 0, 188, 209, 11, 23, 238, 195, 180, 0\n219, 191, 225, 210, 19
8, 198, 206, 9, 182, 195\n214, 211, 210]'),
 Text(2946.2400000000002, 1585.5, 'gini = 0.0\nsamples = 137\nvalue = [21
8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(3303.36, 1585.5, 'X[6] <= 0.74\ngini = 0.942\nsamples = 2231\nvalue
= [0, 0, 0, 188, 209, 11, 23, 238, 195, 180, 0, 219\n191, 225, 210, 198, 1
98, 206, 9, 182, 195, 214\n211, 210]'),
 Text(2856.96, 1132.5, 'X[10] <= 1.173\ngini = 0.86\nsamples = 890\nvalue
= [0, 0, 0, 0, 0, 11, 0, 0, 0, 179, 0, 0, 188\n225, 209, 0, 195, 0, 5, 0,
195, 0, 0, 200]'),
 Text(2678.4, 679.5, 'X[10] <= 0.549\ngini = 0.837\nsamples = 768\nvalue =
```