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 [425]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\madrid_2011.csv")
a
```

Out[425]:

	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	Na
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	NaN	3.8	NaN	96.0	99.0	NaN	NaN	NaN	NaN	NaN	7.
3	2011-11-01 01:00:00	NaN	0.6	NaN	NaN	60.0	83.0	2.0	NaN	NaN	NaN	NaN	Na
4	2011-11-01 01:00:00	NaN	NaN	NaN	NaN	44.0	62.0	3.0	NaN	NaN	3.0	NaN	Na
...	...	...	...	...	...	...	...	...	...	...	...	...	..
209923	2011-09-01 00:00:00	NaN	0.2	NaN	NaN	5.0	19.0	44.0	NaN	NaN	NaN	NaN	Na
209924	2011-09-01 00:00:00	NaN	0.1	NaN	NaN	6.0	29.0	NaN	11.0	NaN	7.0	NaN	Na
209925	2011-09-01 00:00:00	NaN	NaN	NaN	0.23	1.0	21.0	28.0	NaN	NaN	NaN	1.44	Na
209926	2011-09-01 00:00:00	NaN	NaN	NaN	NaN	3.0	15.0	48.0	NaN	NaN	NaN	NaN	Na
209927	2011-09-01 00:00:00	NaN	NaN	NaN	NaN	4.0	33.0	38.0	13.0	NaN	NaN	NaN	Na

209928 rows × 14 columns

In [426]:

```
a.info()
```

```
<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 [427]:

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

Out[427]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH
0	2011-11-01 01:00:00	104.0	1.0	104.0	104.00	154.0	84.0	104.0	104.0	104.0	6.0	104.00
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	104.0	3.8	104.00	96.0	99.0	104.0	104.0	104.0	104.0	104.00
3	2011-11-01 01:00:00	104.0	0.6	104.0	104.00	60.0	83.0	2.0	104.0	104.0	104.0	104.00
4	2011-11-01 01:00:00	104.0	104.0	104.0	104.00	44.0	62.0	3.0	104.0	104.0	3.0	104.00
...	...	...	...	...	...	...	...	...	...	...	...	..
209923	2011-09-01 00:00:00	104.0	0.2	104.0	104.00	5.0	19.0	44.0	104.0	104.0	104.0	104.00
209924	2011-09-01 00:00:00	104.0	0.1	104.0	104.00	6.0	29.0	104.0	11.0	104.0	7.0	104.00
209925	2011-09-01 00:00:00	104.0	104.0	104.0	0.23	1.0	21.0	28.0	104.0	104.0	104.0	1.44
209926	2011-09-01 00:00:00	104.0	104.0	104.0	104.00	3.0	15.0	48.0	104.0	104.0	104.0	104.00
209927	2011-09-01 00:00:00	104.0	104.0	104.0	104.00	4.0	33.0	38.0	13.0	104.0	104.0	104.00

209928 rows × 14 columns

In [428]:

```
b.columns
```

Out[428]:

```
Index(['date', 'BEN', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'O_3', 'PM10', 'P
M25',
       'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [429]:

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

Out[429]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TC
0	2011-11-01 01:00:00	104.0	1.0	104.0	104.00	154.0	84.0	104.0	104.0	104.0	6.0	104.00	104
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	104.0	3.8	104.00	96.0	99.0	104.0	104.0	104.0	104.0	104.00	7
3	2011-11-01 01:00:00	104.0	0.6	104.0	104.00	60.0	83.0	2.0	104.0	104.0	104.0	104.00	104
4	2011-11-01 01:00:00	104.0	104.0	104.0	104.00	44.0	62.0	3.0	104.0	104.0	3.0	104.00	104
5	2011-11-01 01:00:00	0.5	0.8	0.3	104.00	102.0	75.0	2.0	35.0	104.0	5.0	104.00	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	104.0	104.0	104.0	0.36	83.0	78.0	6.0	104.0	104.0	104.0	1.80	104
8	2011-11-01 01:00:00	104.0	0.7	104.0	104.00	80.0	91.0	5.0	104.0	104.0	8.0	104.00	104
9	2011-11-01 01:00:00	104.0	0.6	104.0	104.00	63.0	71.0	104.0	33.0	104.0	6.0	104.00	104

In [443]:

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

Out[443]:

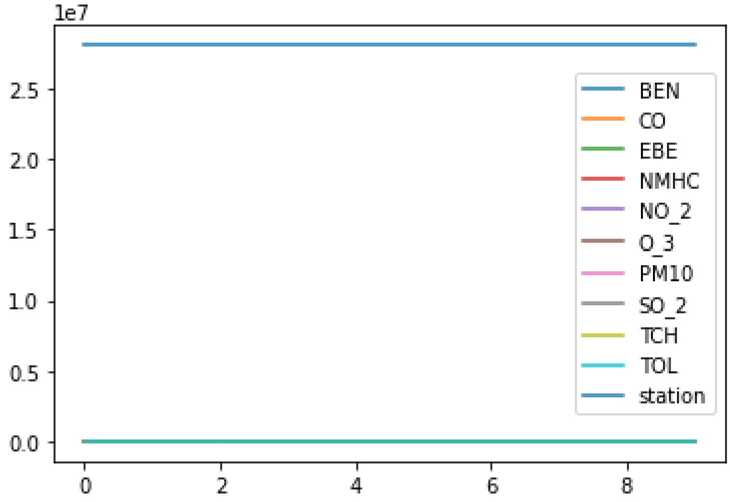
	BEN	CO	EBE	NMHC	NO_2	O_3	PM10	SO_2	TCH	TOL	station
0	104.0	1.0	104.0	104.00	84.0	104.0	104.0	6.0	104.00	104.0	28079004
1	2.5	0.4	3.5	0.26	92.0	3.0	40.0	9.0	1.54	8.7	28079008
2	2.9	104.0	3.8	104.00	99.0	104.0	104.0	104.0	104.00	7.2	28079011
3	104.0	0.6	104.0	104.00	83.0	2.0	104.0	104.0	104.00	104.0	28079016
4	104.0	104.0	104.0	104.00	62.0	3.0	104.0	3.0	104.00	104.0	28079017
5	0.5	0.8	0.3	104.00	75.0	2.0	35.0	5.0	104.00	4.3	28079018
6	0.7	0.3	1.1	0.16	66.0	7.0	22.0	2.0	1.36	1.7	28079024
7	104.0	104.0	104.0	0.36	78.0	6.0	104.0	104.0	1.80	104.0	28079027
8	104.0	0.7	104.0	104.00	91.0	5.0	104.0	8.0	104.00	104.0	28079035
9	104.0	0.6	104.0	104.00	71.0	104.0	33.0	6.0	104.00	104.0	28079036

In [444]:

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

Out[444]:

```
<AxesSubplot:>
```

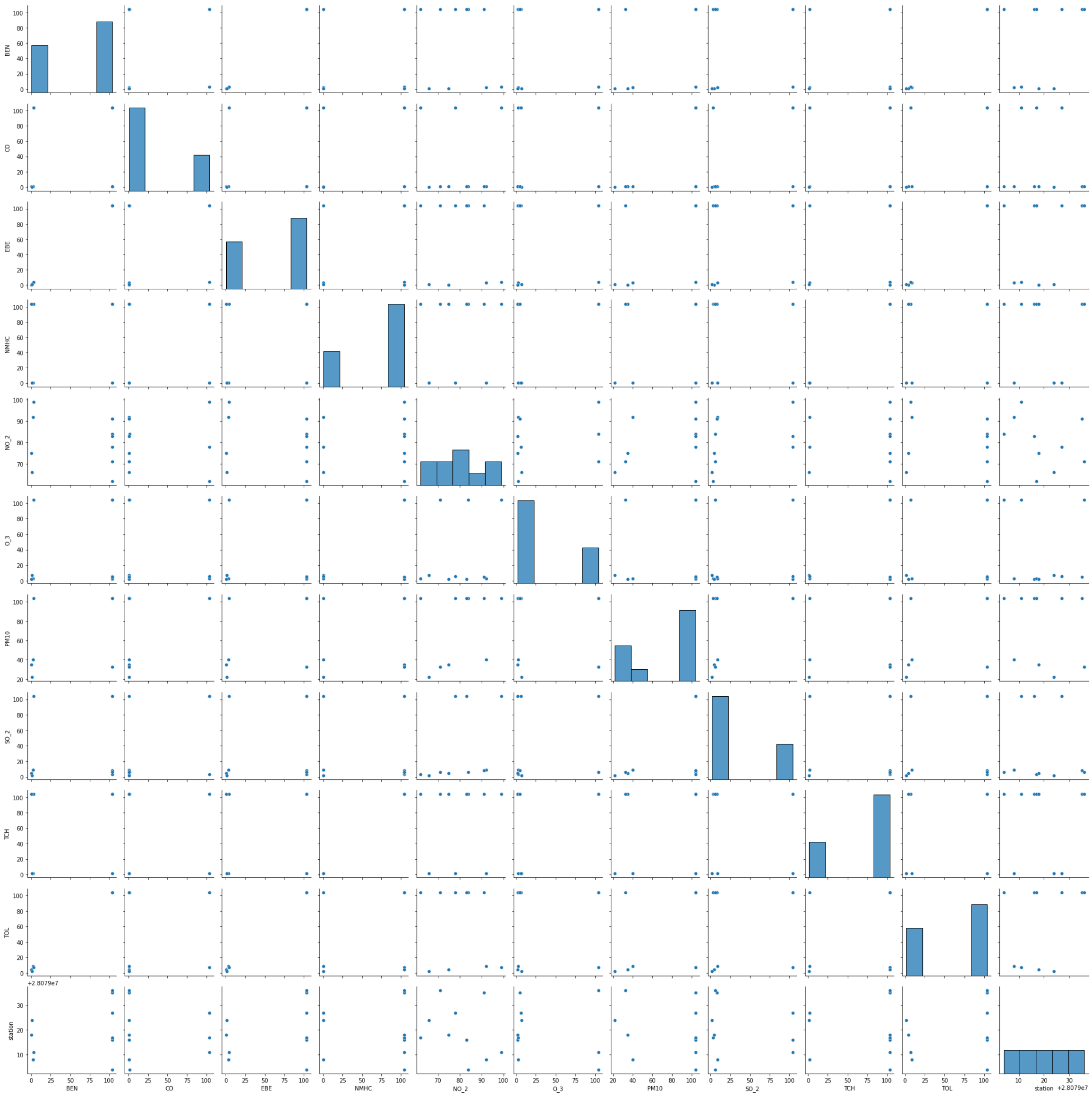


In [445]:

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

Out[445]:

```
<seaborn.axisgrid.PairGrid at 0x118560e6070>
```



In [447]:

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

In [448]:

```
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 [449]:

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

Out[449]:

```
LinearRegression()
```

In [450]:

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

```
1.1475407183659598
```

In [451]:

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

Out[451]:

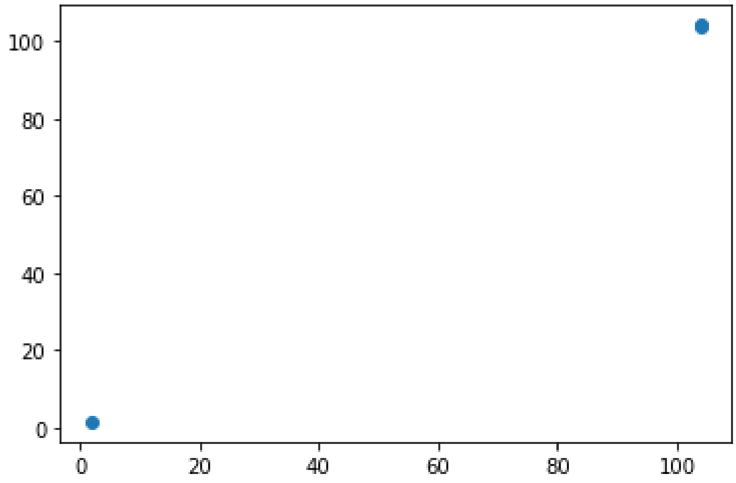
	Co-efficient
BEN	-1.361791e-01
CO	-2.969105e-16
EBE	1.359164e-01
NMHC	9.892286e-01
NO_2	5.820232e-17

In [452]:

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

Out[452]:

```
<matplotlib.collections.PathCollection at 0x11861bfde50>
```



In [453]:

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

```
0.9999817735529777
```

In [454]:

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

In [455]:

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

Out[455]:

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

In [456]:

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

Out[456]:

```
0.9999995112806245
```

In [457]:

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

Out[457]:

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

In [458]:

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

Out[458]:

```
0.999992167982922
```

In [459]:

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

Out[459]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH
0	2011-11-01 01:00:00	104.0	1.0	104.0	104.00	154.0	84.0	104.0	104.0	104.0	6.0	104.00
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	104.0	3.8	104.00	96.0	99.0	104.0	104.0	104.0	104.0	104.00
3	2011-11-01 01:00:00	104.0	0.6	104.0	104.00	60.0	83.0	2.0	104.0	104.0	104.0	104.00
4	2011-11-01 01:00:00	104.0	104.0	104.0	104.00	44.0	62.0	3.0	104.0	104.0	3.0	104.00
...	...	...	...	...	...	...	...	...	...	...	...	...
6995	2011-11-13 04:00:00	104.0	0.2	104.0	104.00	1.0	17.0	50.0	104.0	104.0	104.0	104.00
6996	2011-11-13 04:00:00	104.0	104.0	104.0	104.00	1.0	6.0	104.0	11.0	104.0	1.0	104.00
6997	2011-11-13 04:00:00	104.0	104.0	104.0	104.00	1.0	14.0	104.0	12.0	8.0	104.0	104.00
6998	2011-11-13 04:00:00	104.0	104.0	104.0	104.00	2.0	16.0	104.0	8.0	4.0	104.0	104.00
6999	2011-11-13 04:00:00	104.0	104.0	104.0	104.00	1.0	8.0	57.0	104.0	104.0	104.0	104.00

7000 rows × 14 columns

In [469]:

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

In [470]:

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

In [471]:

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

In [472]:

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

Out[472]:

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

In [473]:

```
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 [481]:

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

In [482]:

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

```
[28079050]
```

In [483]:

```
logr.classes_
```

Out[483]:

```
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 [484]:

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

Out[484]:

```
0.0
```

In [485]:

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

Out[485]:

```
0.0
```

In [486]:

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

Out[486]:

```
0.99
```

In [487]:

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

Out[487]:

```
ElasticNet()
```

In [488]:

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

```
[2.97089556e-04 0.00000000e+00 0.00000000e+00 9.87367208e-01
 0.00000000e+00]
```

In [489]:

```
print(en.intercept_)

```

```
1.2756670298839623
```

In [490]:

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

```
0.9999970585682351
```

In [491]:

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

Out[491]:

```
RandomForestClassifier()
```

In [492]:

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

In [493]:

```
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[493]:

```
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 [494]:

```
grid_search.best_score_
```

Out[494]:

```
0.9989795918367347
```

In [495]:

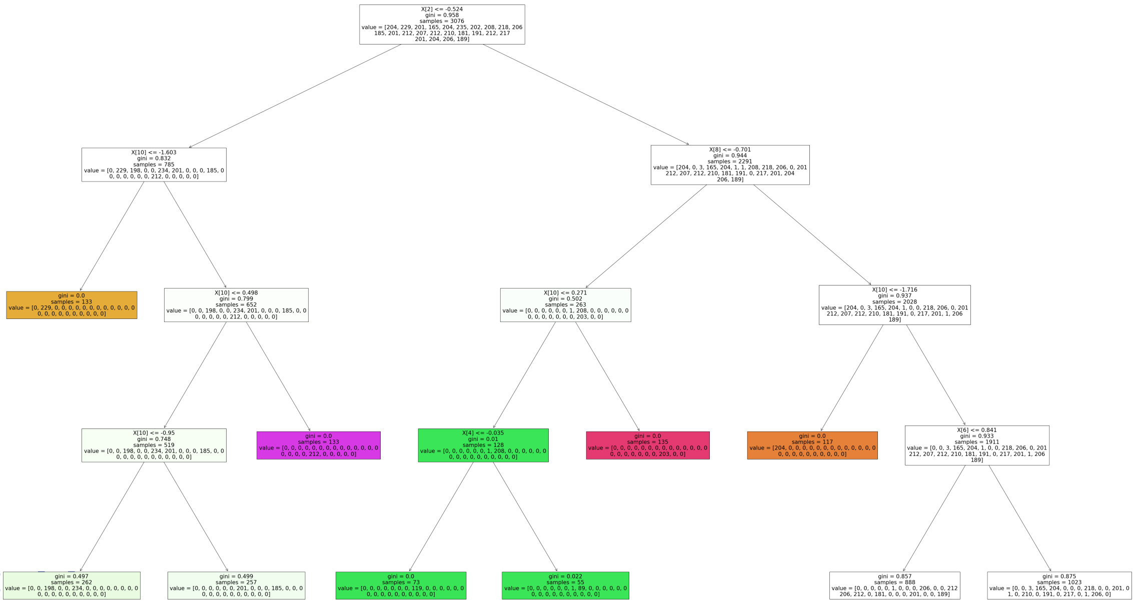
```
rfc_best=grid_search.best_estimator_
```

In [480]:

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

Out[480]:

```
[Text(1753.7142857142856, 2446.2, 'X[2] <= -0.524\ngini = 0.958\nsamples =
3076\nvalue = [204, 229, 201, 165, 204, 235, 202, 208, 218, 206\n185, 201,
212, 207, 212, 210, 181, 191, 212, 217\n201, 204, 206, 189]'),
 Text(637.7142857142857, 1902.6, 'X[10] <= -1.603\ngini = 0.832\nsamples =
785\nvalue = [0, 229, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0\n0, 0, 0, 0, 0,
0, 212, 0, 0, 0, 0, 0]'),
 Text(318.85714285714283, 1359.0, 'gini = 0.0\nsamples = 133\nvalue = [0,
229, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(956.5714285714284, 1359.0, 'X[10] <= 0.498\ngini = 0.799\nsamples =
652\nvalue = [0, 0, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0, 0\n0, 0, 0, 0,
0, 212, 0, 0, 0, 0, 0]'),
 Text(637.7142857142857, 815.3999999999999, 'X[10] <= -0.95\ngini = 0.748
\nsamples = 519\nvalue = [0, 0, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0, 0\n
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(318.85714285714283, 271.7999999999997, 'gini = 0.497\nsamples = 262
\nvalue = [0, 0, 198, 0, 0, 234, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(956.5714285714284, 271.7999999999997, 'gini = 0.499\nsamples = 257\n
value = [0, 0, 0, 0, 0, 0, 201, 0, 0, 0, 185, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(1275.4285714285713, 815.3999999999999, 'gini = 0.0\nsamples = 133\nv
alue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 212, 0, 0,
0, 0, 0]'),
 Text(2869.7142857142853, 1902.6, 'X[8] <= -0.701\ngini = 0.944\nsamples =
2291\nvalue = [204, 0, 3, 165, 204, 1, 1, 208, 218, 206, 0, 201\n212, 207,
212, 210, 181, 191, 0, 217, 201, 204\n206, 189]'),
 Text(2232.0, 1359.0, 'X[10] <= 0.271\ngini = 0.502\nsamples = 263\nvalue
= [0, 0, 0, 0, 0, 0, 1, 208, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 203,
0, 0]'),
 Text(1913.1428571428569, 815.3999999999999, 'X[4] <= -0.035\ngini = 0.01
\nsamples = 128\nvalue = [0, 0, 0, 0, 0, 0, 1, 208, 0, 0, 0, 0, 0, 0\n0,
0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1594.2857142857142, 271.7999999999997, 'gini = 0.0\nsamples = 73\nva
lue = [0, 0, 0, 0, 0, 0, 0, 119, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(2232.0, 271.7999999999997, 'gini = 0.022\nsamples = 55\nvalue = [0,
0, 0, 0, 0, 0, 1, 89, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2550.8571428571427, 815.3999999999999, 'gini = 0.0\nsamples = 135\nv
alue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 20
3, 0, 0]'),
 Text(3507.428571428571, 1359.0, 'X[10] <= -1.716\ngini = 0.937\nsamples =
2028\nvalue = [204, 0, 3, 165, 204, 1, 0, 0, 218, 206, 0, 201\n212, 207, 2
12, 210, 181, 191, 0, 217, 201, 1, 206\n189]'),
 Text(3188.5714285714284, 815.3999999999999, 'gini = 0.0\nsamples = 117\nv
alue = [204, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(3826.2857142857138, 815.3999999999999, 'X[6] <= 0.841\ngini = 0.933
\nsamples = 1911\nvalue = [0, 0, 3, 165, 204, 1, 0, 0, 218, 206, 0, 201\n2
12, 207, 212, 210, 181, 191, 0, 217, 201, 1, 206\n189]'),
 Text(3507.428571428571, 271.7999999999997, 'gini = 0.857\nsamples = 888\n
value = [0, 0, 0, 0, 0, 1, 0, 0, 0, 206, 0, 0, 212\n206, 212, 0, 181, 0,
0, 0, 201, 0, 0, 189]'),
 Text(4145.142857142857, 271.7999999999997, 'gini = 0.875\nsamples = 1023
\nvalue = [0, 0, 3, 165, 204, 0, 0, 0, 218, 0, 0, 201, 0\n1, 0, 210, 0, 19
1, 0, 217, 0, 1, 206, 0]')]
```


X[2] <= -0.524 gini = 0.958 samples = 3076 value = [204, 229, 201, 165, 204, 235, 202, 208, 218, 206 185, 201, 212, 207, 212, 210, 181, 191, 212, 217 201, 204, 206, 189]
X[10] <= -1.603 gini = 0.832 samples = 785 value = [0, 229, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0 0, 0, 0, 0, 0, 0, 212, 0, 0, 0, 0, 0]
X[8] <= -0.701 gini = 0.944 samples = 2291 value = [204, 0, 3, 165, 204, 1, 1, 208, 218, 206, 0, 201 212, 207, 212, 210, 181, 191, 0, 217, 201, 204 206, 189]
gini = 0.0 samples = 133 value = [0, 229, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
X[10] <= 0.498 gini = 0.799 samples = 652 value = [0, 0, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0, 0 0, 0, 0, 0, 0, 212, 0, 0, 0, 0, 0]
X[10] <= 0.271 gini = 0.502 samples = 263 value = [0, 0, 0, 0, 0, 0, 1, 208, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 203, 0, 0]
X[10] <= -1.716 gini = 0.937 samples = 2028 value = [204, 0, 3, 165, 204, 1, 0, 0, 218, 206, 0, 201 212, 207, 212, 210, 181, 191, 0, 217, 201, 1, 206 189]
X[10] <= -0.95 gini = 0.748 samples = 519 value = [0, 0, 198, 0, 0, 234, 201, 0, 0, 0, 185, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.0 samples = 133 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 212, 0, 0, 0, 0, 0]
X[4] <= -0.035 gini = 0.01 samples = 128 value = [0, 0, 0, 0, 0, 0, 1, 208, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.0 samples = 135 value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 203, 0, 0]
gini = 0.0 samples = 117 value = [204, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
X[6] <= 0.841 gini = 0.933 samples = 1911 value = [0, 0, 3, 165, 204, 1, 0, 0, 218, 206, 0, 201 212, 207, 212, 210, 181, 191, 0, 217, 201, 1, 206 189]
gini = 0.497 samples = 262 value = [0, 0, 198, 0, 0, 234, 0, 0, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.499 samples = 257 value = [0, 0, 0, 0, 0, 0, 201, 0, 0, 0, 185, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.0 samples = 73 value = [0, 0, 0, 0, 0, 0, 0, 119, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.022 samples = 55 value = [0, 0, 0, 0, 0, 0, 1, 89, 0, 0, 0, 0, 0, 0 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
gini = 0.857 samples = 888 value = [0, 0, 0, 0, 0, 1, 0, 0, 0, 206, 0, 0, 212 206, 212, 0, 181, 0, 0, 0, 201, 0, 0, 189]
gini = 0.875 samples = 1023 value = [0, 0, 3, 165, 204, 0, 0, 0, 218, 0, 0, 201, 0 1, 0, 210, 0, 191, 0, 217, 0, 1, 206, 0]