2009 ¶

In [1]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression,LogisticRegression,Lasso,Rid
from sklearn.model_selection import train_test_split
```

In [2]:
```
df=pd.read_csv("madrid_2009.csv")
df
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM
0	2009-10-01 01:00:00	NaN	0.27	NaN	NaN	NaN	39.889999	48.150002	NaN	50.680000	18.260(
1	2009-10-01 01:00:00	NaN	0.22	NaN	NaN	NaN	21.230000	24.260000	NaN	55.880001	10.580(
2	2009-10-01 01:00:00	NaN	0.18	NaN	NaN	NaN	31.230000	34.880001	NaN	49.060001	25.190(
3	2009-10-01 01:00:00	0.95	0.33	1.43	2.68	0.25	55.180000	81.360001	1.57	36.669998	26.530(
4	2009-10-01 01:00:00	NaN	0.41	NaN	NaN	0.12	61.349998	76.260002	NaN	38.090000	23.760(
...	...	...	...	...	...	...	...	...	...	...	
215683	2009-06-01 00:00:00	0.50	0.22	0.39	0.75	0.09	22.000000	24.510000	1.00	82.239998	10.830(
215684	2009-06-01 00:00:00	NaN	0.31	NaN	NaN	NaN	76.110001	101.099998	NaN	41.220001	9.920(
215685	2009-06-01 00:00:00	0.13	NaN	0.86	NaN	0.23	81.050003	99.849998	NaN	24.830000	12.460(
215686	2009-06-01 00:00:00	0.21	NaN	2.96	NaN	0.10	72.419998	82.959999	NaN	NaN	13.030(
215687	2009-06-01 00:00:00	0.37	0.32	0.99	1.36	0.14	54.290001	64.480003	1.06	56.919998	15.360(

215688 rows × 17 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215688 entries, 0 to 215687
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     215688 non-null  object 
 1   BEN      60082 non-null   float64
 2   CO       190801 non-null  float64
 3   EBE      60081 non-null   float64
 4   MXY      24846 non-null   float64
 5   NMHC     74748 non-null   float64
 6   NO_2     214562 non-null  float64
 7   NOx      214565 non-null  float64
 8   OXY      24854 non-null   float64
 9   O_3      204482 non-null  float64
 10  PM10     196331 non-null  float64
 11  PM25     55822 non-null   float64
 12  PXY      24854 non-null   float64
 13  SO_2     212671 non-null  float64
 14  TCH      75213 non-null   float64
 15  TOL      59920 non-null   float64
 16  station  215688 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 28.0+ MB
```

In [4]:
```
df1=df.dropna()
df1
```

Out[4]:

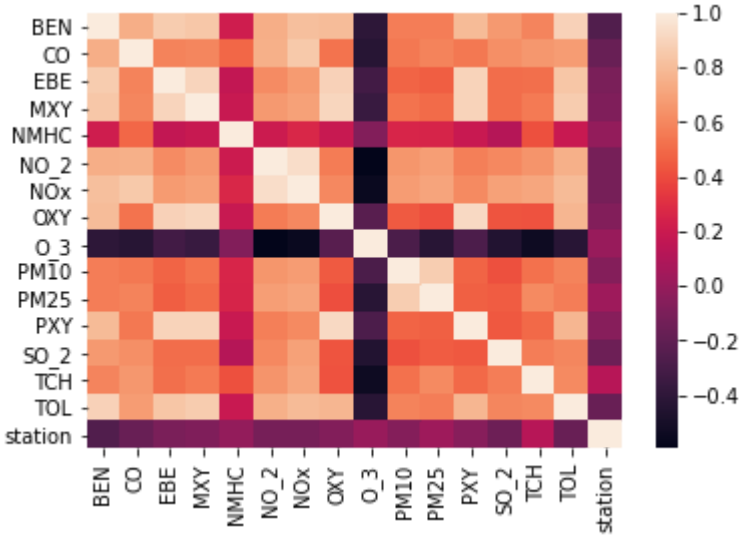
	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM1
3	2009-10-01 01:00:00	0.95	0.33	1.43	2.68	0.25	55.180000	81.360001	1.57	36.669998	26.5300(
20	2009-10-01 01:00:00	0.38	0.32	0.32	0.89	0.01	17.969999	19.240000	1.00	65.870003	10.5200(
24	2009-10-01 01:00:00	0.55	0.24	0.65	1.79	0.18	36.619999	43.919998	1.28	48.070000	19.1500(
28	2009-10-01 02:00:00	0.65	0.21	1.20	2.04	0.18	37.169998	48.869999	1.21	26.950001	32.2000(
45	2009-10-01 02:00:00	0.38	0.30	0.50	1.15	0.00	17.889999	19.299999	1.00	60.009998	12.2600(
...	...	...	...	...	...	...	...	...	...	...	
215659	2009-05-31 23:00:00	0.54	0.27	1.00	0.69	0.09	28.280001	29.490000	0.86	78.750000	15.1700(
215663	2009-05-31 23:00:00	0.74	0.35	1.13	1.65	0.15	56.410000	69.870003	1.26	56.799999	11.8000(
215667	2009-06-01 00:00:00	0.78	0.29	0.99	1.96	0.04	64.870003	82.629997	1.13	58.000000	12.6700(
215683	2009-06-01 00:00:00	0.50	0.22	0.39	0.75	0.09	22.000000	24.510000	1.00	82.239998	10.8300(
215687	2009-06-01 00:00:00	0.37	0.32	0.99	1.36	0.14	54.290001	64.480003	1.06	56.919998	15.3600(

24717 rows × 17 columns

In [5]:
```
df1=df1.drop(["date"],axis=1)
```

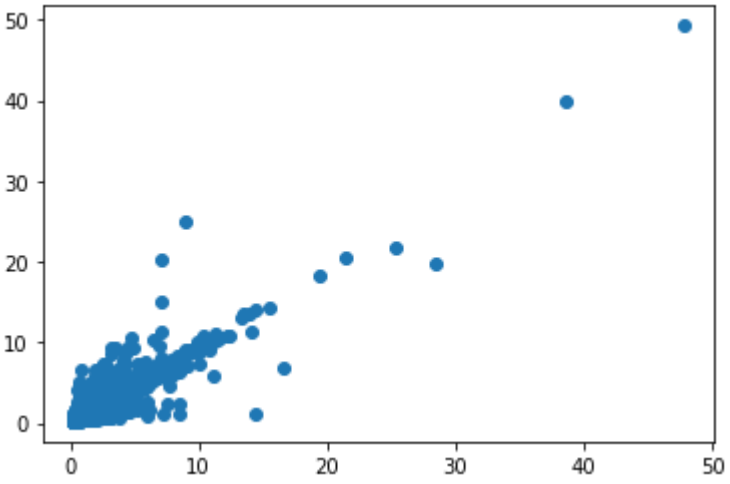
In [6]:
```
sns.heatmap(df1.corr())
```

Out[6]: <AxesSubplot:>



In [7]:
```
plt.plot(df1["EBE"],df1["PXY"],"o")
```

Out[7]: [<matplotlib.lines.Line2D at 0x23c87847160>]



In [8]:
```
data=df[["EBE","PXY"]]
```

In [9]:
```
# sns.stripplot(x=df["EBE"],y=df["PXY"],jitter=True,marker='o',color='blue')
```

In [10]:
```
x=df1.drop(["EBE"],axis=1)
y=df1["EBE"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

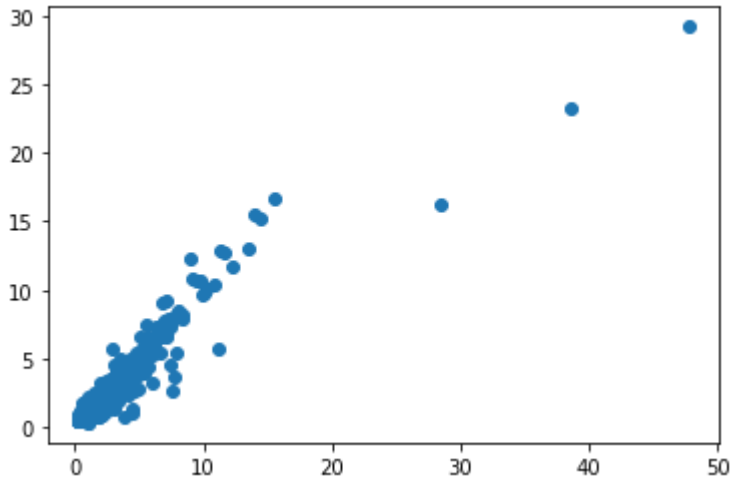
Linear

In [11]:
```
li=LinearRegression()
li.fit(x_train,y_train)
```

Out[11]:
```
LinearRegression()
```

In [12]:
```
prediction=li.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[12]:
```
<matplotlib.collections.PathCollection at 0x23c8790c130>
```



In [13]:
```
lis=li.score(x_test,y_test)
```

In [14]:
```
df1["TCH"].value_counts()
```

Out[14]:
```
1.39    1091
1.36    1056
1.38    1046
1.40    1018
1.37    1017
        ...
2.52       1
1.16       1
2.41       1
1.13       1
2.79       1
Name: TCH, Length: 169, dtype: int64
```

In [15]:
```
df1.loc[df1["TCH"]<1.40,"TCH"]=1
df1.loc[df1["TCH"]>1.40,"TCH"]=2
df1["TCH"].value_counts()
```

Out[15]:
```
1.0    12963
2.0    11754
Name: TCH, dtype: int64
```

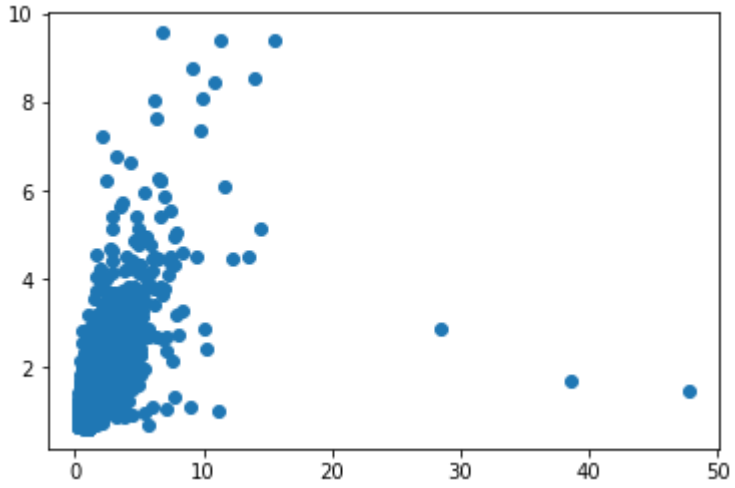
Lasso

In [16]:
```
la=Lasso(alpha=5)
la.fit(x_train,y_train)
```

Out[16]:
```
Lasso(alpha=5)
```

In [17]:
```
prediction1=la.predict(x_test)
plt.scatter(y_test,prediction1)
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x23c88533880>
```



In [18]:
```
las=la.score(x_test,y_test)
```

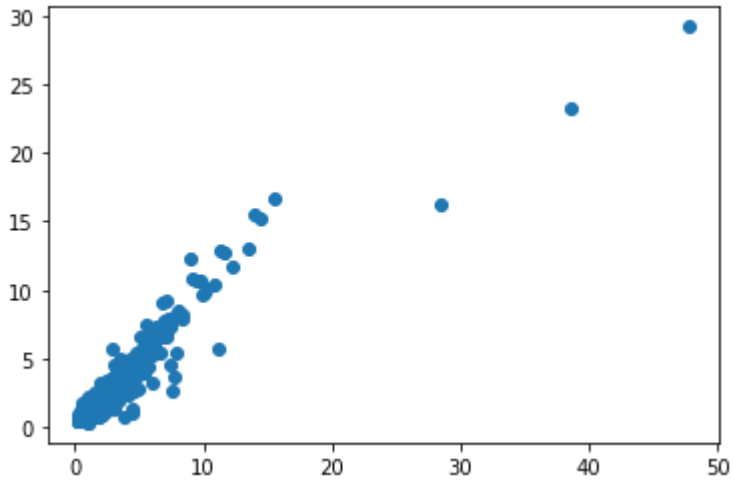
Ridge

In [19]:
```
rr=Ridge(alpha=1)
rr.fit(x_train,y_train)
```

Out[19]:
```
Ridge(alpha=1)
```

In [20]:
```
prediction2=rr.predict(x_test)
plt.scatter(y_test,prediction2)
```

Out[20]: <matplotlib.collections.PathCollection at 0x23c871da250>



In [21]:
```
rrs=rr.score(x_test,y_test)
```

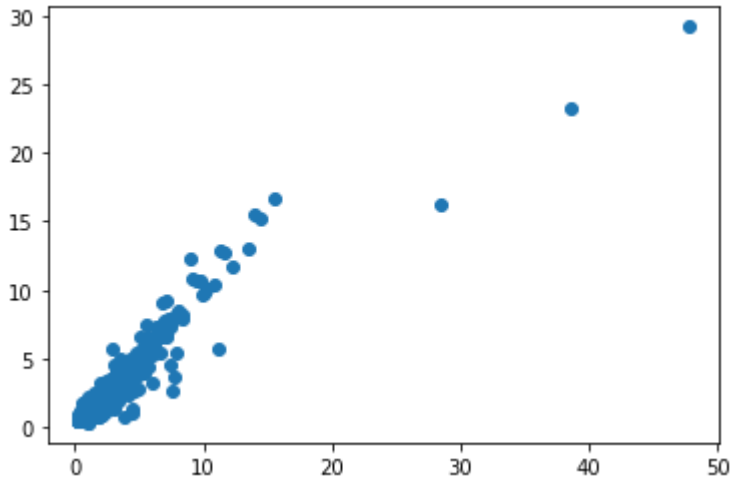
ElasticNet

In [22]:
```
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[22]: ElasticNet()

In [23]:
```
prediction2=rr.predict(x_test)
plt.scatter(y_test,prediction2)
```

Out[23]: <matplotlib.collections.PathCollection at 0x23c885beca0>



In [24]:
```
ens=en.score(x_test,y_test)
```

In [25]:
```
print(rr.score(x_test,y_test))
rr.score(x_train,y_train)
```

```
0.8783497693150732
```

Out[25]:
```
0.8864182797143654
```

Logistic

In [26]:
```
g={"TCH":{1.0:"Low",2.0:"High"}}
df1=df1.replace(g)
df1["TCH"].value_counts()
```

Out[26]:
```
Low     12963
High    11754
Name: TCH, dtype: int64
```

In [27]:
```
x=df1.drop(["TCH"],axis=1)
y=df1["TCH"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [28]:
```
lo=LogisticRegression()
lo.fit(x_train,y_train)
```

Out[28]:
```
LogisticRegression()
```

In [29]:
```
prediction3=lo.predict(x_test)
plt.scatter(y_test,prediction3)
```

Out[29]:
```
<matplotlib.collections.PathCollection at 0x23c88602520>
```

In [30]:
```
los=lo.score(x_test,y_test)
```

Random Forest

In [31]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
```

In [32]:
```
g1={"TCH":{"Low":1.0,"High":2.0}}
df1=df1.replace(g1)
```

In [33]:
```
x=df1.drop(["TCH"],axis=1)
y=df1["TCH"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [34]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

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

In [35]:
```
parameter={
    'max_depth':[1,2,4,5,6],
    'min_samples_leaf':[5,10,15,20,25],
    'n_estimators':[10,20,30,40,50]
}
```

In [36]:
```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameter,cv=2,scoring="accu
grid_search.fit(x_train,y_train)
```

Out[36]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 4, 5, 6],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [37]:
```
rfcs=grid_search.best_score_
```

In [38]:
```
rfc_best=grid_search.best_estimator_
```

In [39]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes',"
```

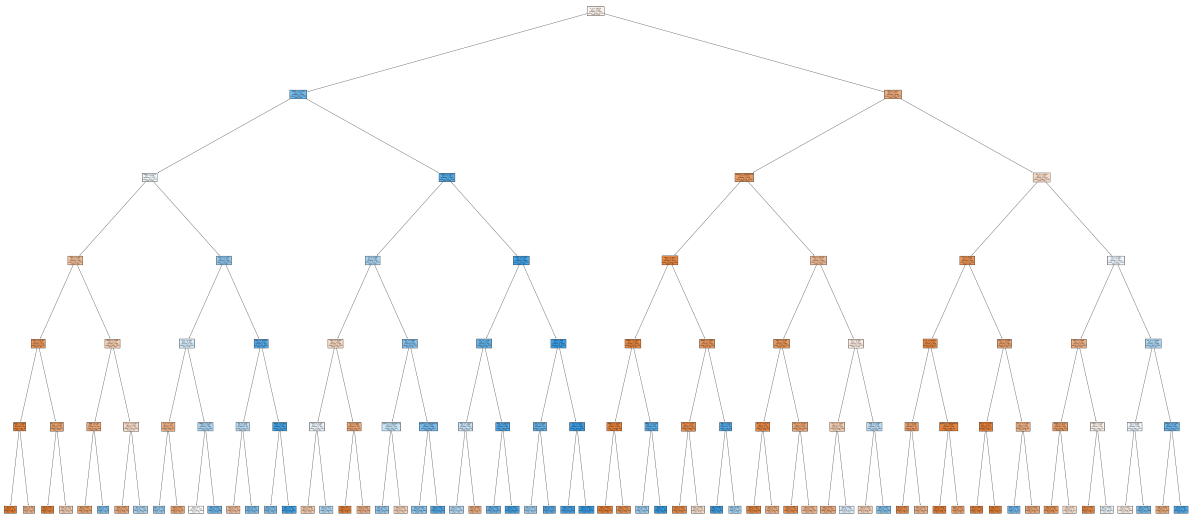
Out[39]:
```
[Text(2232.0, 2019.0857142857144, 'O_3 <= 36.105\ngini = 0.499\nsamples = 109
31\nvalue = [9061, 8240]\nclass = Yes'),
 Text(1116.0, 1708.457142857143, 'PM25 <= 10.105\ngini = 0.35\nsamples = 4675
\nvalue = [1681, 5753]\nclass = No'),
 Text(558.0, 1397.8285714285716, 'NMHC <= 0.175\ngini = 0.499\nsamples = 1175
\nvalue = [912, 974]\nclass = No'),
 Text(279.0, 1087.2, 'EBE <= 0.585\ngini = 0.445\nsamples = 580\nvalue = [62
0, 311]\nclass = Yes'),
 Text(139.5, 776.5714285714287, 'NOx <= 37.82\ngini = 0.262\nsamples = 116\nv
alue = [153, 28]\nclass = Yes'),
 Text(69.75, 465.9428571428573, 'BEN <= 0.565\ngini = 0.104\nsamples = 46\nva
lue = [69, 4]\nclass = Yes'),
 Text(34.875, 155.3142857142857, 'gini = 0.0\nsamples = 36\nvalue = [60, 0]\n
class = Yes'),
 Text(104.625, 155.3142857142857, 'gini = 0.426\nsamples = 10\nvalue = [9, 4]
\nclass = Yes'),
 Text(209.25, 465.9428571428573, 'SO_2 <= 10.58\ngini = 0.346\nsamples = 70\n
value = [84, 24]\nclass = Yes'),
 Text(174.375, 155.3142857142857, 'gini = 0.041\nsamples = 31\nvalue = [47,
1]\nclass = Yes'),
 Text(244.125, 155.3142857142857, 'gini = 0.473\nsamples = 39\nvalue = [37, 2
3]\nclass = Yes'),
 Text(418.5, 776.5714285714287, 'PM25 <= 5.985\ngini = 0.47\nsamples = 464\nv
alue = [467, 283]\nclass = Yes'),
 Text(348.75, 465.9428571428573, 'BEN <= 1.65\ngini = 0.371\nsamples = 127\nv
alue = [150, 49]\nclass = Yes'),
 Text(313.875, 155.3142857142857, 'gini = 0.341\nsamples = 117\nvalue = [147,
41]\nclass = Yes'),
 Text(383.625, 155.3142857142857, 'gini = 0.397\nsamples = 10\nvalue = [3, 8]
\nclass = No'),
 Text(488.25, 465.9428571428573, 'SO_2 <= 10.84\ngini = 0.489\nsamples = 337
\nvalue = [317, 234]\nclass = Yes'),
 Text(453.375, 155.3142857142857, 'gini = 0.367\nsamples = 179\nvalue = [222,
71]\nclass = Yes'),
 Text(523.125, 155.3142857142857, 'gini = 0.465\nsamples = 158\nvalue = [95,
163]\nclass = No'),
 Text(837.0, 1087.2, 'MXY <= 1.475\ngini = 0.425\nsamples = 595\nvalue = [29
2, 663]\nclass = No'),
 Text(697.5, 776.5714285714287, 'SO_2 <= 6.63\ngini = 0.493\nsamples = 334\nv
alue = [234, 295]\nclass = No'),
 Text(627.75, 465.9428571428573, 'O_3 <= 5.19\ngini = 0.402\nsamples = 72\nva
lue = [83, 32]\nclass = Yes'),
 Text(592.875, 155.3142857142857, 'gini = 0.444\nsamples = 10\nvalue = [4, 8]
\nclass = No'),
 Text(662.625, 155.3142857142857, 'gini = 0.357\nsamples = 62\nvalue = [79, 2
4]\nclass = Yes'),
 Text(767.25, 465.9428571428573, 'NMHC <= 0.225\ngini = 0.463\nsamples = 262
\nvalue = [151, 263]\nclass = No'),
 Text(732.375, 155.3142857142857, 'gini = 0.5\nsamples = 136\nvalue = [107, 1
05]\nclass = Yes'),
 Text(802.125, 155.3142857142857, 'gini = 0.341\nsamples = 126\nvalue = [44,
158]\nclass = No'),
 Text(976.5, 776.5714285714287, 'NOx <= 64.325\ngini = 0.235\nsamples = 261\n
value = [58, 368]\nclass = No'),
 Text(906.75, 465.9428571428573, 'TOL <= 2.44\ngini = 0.479\nsamples = 59\nva
lue = [35, 53]\nclass = No'),
 Text(871.875, 155.3142857142857, 'gini = 0.444\nsamples = 18\nvalue = [20, 1
```

```
0]\nclass = Yes'),
 Text(941.625, 155.3142857142857, 'gini = 0.383\nsamples = 41\nvalue = [15, 4
3]\nclass = No'),
 Text(1046.25, 465.9428571428573, 'PM10 <= 6.485\ngini = 0.127\nsamples = 202
\nvalue = [23, 315]\nclass = No'),
 Text(1011.375, 155.3142857142857, 'gini = 0.391\nsamples = 10\nvalue = [4, 1
1]\nclass = No'),
 Text(1081.125, 155.3142857142857, 'gini = 0.111\nsamples = 192\nvalue = [19,
304]\nclass = No'),
 Text(1674.0, 1397.8285714285716, 'NMHC <= 0.175\ngini = 0.239\nsamples = 350
0\nvalue = [769, 4779]\nclass = No'),
 Text(1395.0, 1087.2, 'CO <= 0.385\ngini = 0.465\nsamples = 871\nvalue = [51
0, 875]\nclass = No'),
 Text(1255.5, 776.5714285714287, 'NOx <= 81.99\ngini = 0.483\nsamples = 259\n
value = [240, 166]\nclass = Yes'),
 Text(1185.75, 465.9428571428573, 'MXY <= 1.325\ngini = 0.499\nsamples = 156
\nvalue = [116, 128]\nclass = No'),
 Text(1150.875, 155.3142857142857, 'gini = 0.466\nsamples = 59\nvalue = [58,
34]\nclass = Yes'),
 Text(1220.625, 155.3142857142857, 'gini = 0.472\nsamples = 97\nvalue = [58,
94]\nclass = No'),
 Text(1325.25, 465.9428571428573, 'CO <= 0.305\ngini = 0.359\nsamples = 103\n
value = [124, 38]\nclass = Yes'),
 Text(1290.375, 155.3142857142857, 'gini = 0.0\nsamples = 18\nvalue = [29, 0]
\nclass = Yes'),
 Text(1360.125, 155.3142857142857, 'gini = 0.408\nsamples = 85\nvalue = [95,
38]\nclass = Yes'),
 Text(1534.5, 776.5714285714287, 'PXY <= 0.805\ngini = 0.399\nsamples = 612\n
value = [270, 709]\nclass = No'),
 Text(1464.75, 465.9428571428573, 'station <= 28079015.0\ngini = 0.487\nsampl
es = 116\nvalue = [81, 112]\nclass = No'),
 Text(1429.875, 155.3142857142857, 'gini = 0.437\nsamples = 73\nvalue = [41,
86]\nclass = No'),
 Text(1499.625, 155.3142857142857, 'gini = 0.478\nsamples = 43\nvalue = [40,
26]\nclass = Yes'),
 Text(1604.25, 465.9428571428573, 'station <= 28079015.0\ngini = 0.365\nsampl
es = 496\nvalue = [189, 597]\nclass = No'),
 Text(1569.375, 155.3142857142857, 'gini = 0.419\nsamples = 337\nvalue = [15
6, 366]\nclass = No'),
 Text(1639.125, 155.3142857142857, 'gini = 0.219\nsamples = 159\nvalue = [33,
231]\nclass = No'),
 Text(1953.0, 1087.2, 'MXY <= 1.595\ngini = 0.117\nsamples = 2629\nvalue = [2
59, 3904]\nclass = No'),
 Text(1813.5, 776.5714285714287, 'SO_2 <= 7.28\ngini = 0.303\nsamples = 620\n
value = [181, 792]\nclass = No'),
 Text(1743.75, 465.9428571428573, 'MXY <= 1.515\ngini = 0.488\nsamples = 125
\nvalue = [91, 124]\nclass = No'),
 Text(1708.875, 155.3142857142857, 'gini = 0.474\nsamples = 111\nvalue = [74,
118]\nclass = No'),
 Text(1778.625, 155.3142857142857, 'gini = 0.386\nsamples = 14\nvalue = [17,
6]\nclass = Yes'),
 Text(1883.25, 465.9428571428573, 'NOx <= 106.45\ngini = 0.209\nsamples = 495
\nvalue = [90, 668]\nclass = No'),
 Text(1848.375, 155.3142857142857, 'gini = 0.319\nsamples = 208\nvalue = [61,
245]\nclass = No'),
 Text(1918.125, 155.3142857142857, 'gini = 0.12\nsamples = 287\nvalue = [29,
423]\nclass = No'),
```

```
 Text(2092.5, 776.5714285714287, 'TOL <= 2.82\ngini = 0.048\nsamples = 2009\n
value = [78, 3112]\nclass = No'),
 Text(2022.75, 465.9428571428573, 'OXY <= 0.765\ngini = 0.314\nsamples = 57\n
value = [16, 66]\nclass = No'),
 Text(1987.875, 155.3142857142857, 'gini = 0.454\nsamples = 14\nvalue = [8, 1
5]\nclass = No'),
 Text(2057.625, 155.3142857142857, 'gini = 0.234\nsamples = 43\nvalue = [8, 5
1]\nclass = No'),
 Text(2162.25, 465.9428571428573, 'TOL <= 4.515\ngini = 0.039\nsamples = 1952
\nvalue = [62, 3046]\nclass = No'),
 Text(2127.375, 155.3142857142857, 'gini = 0.115\nsamples = 310\nvalue = [31,
475]\nclass = No'),
 Text(2197.125, 155.3142857142857, 'gini = 0.024\nsamples = 1642\nvalue = [3
1, 2571]\nclass = No'),
 Text(3348.0, 1708.457142857143, 'BEN <= 0.635\ngini = 0.377\nsamples = 6256
\nvalue = [7380, 2487]\nclass = Yes'),
 Text(2790.0, 1397.8285714285716, 'station <= 28079062.0\ngini = 0.275\nsampl
es = 4122\nvalue = [5420, 1069]\nclass = Yes'),
 Text(2511.0, 1087.2, 'NO_2 <= 29.57\ngini = 0.127\nsamples = 2470\nvalue =
[3648, 266]\nclass = Yes'),
 Text(2371.5, 776.5714285714287, 'NMHC <= 0.655\ngini = 0.086\nsamples = 1970
\nvalue = [2974, 140]\nclass = Yes'),
 Text(2301.75, 465.9428571428573, 'NMHC <= 0.455\ngini = 0.038\nsamples = 190
7\nvalue = [2959, 59]\nclass = Yes'),
 Text(2266.875, 155.3142857142857, 'gini = 0.016\nsamples = 1700\nvalue = [26
62, 22]\nclass = Yes'),
 Text(2336.625, 155.3142857142857, 'gini = 0.197\nsamples = 207\nvalue = [29
7, 37]\nclass = Yes'),
 Text(2441.25, 465.9428571428573, 'NMHC <= 0.725\ngini = 0.264\nsamples = 63
\nvalue = [15, 81]\nclass = No'),
 Text(2406.375, 155.3142857142857, 'gini = 0.449\nsamples = 28\nvalue = [15,
29]\nclass = No'),
 Text(2476.125, 155.3142857142857, 'gini = 0.0\nsamples = 35\nvalue = [0, 52]
\nclass = No'),
 Text(2650.5, 776.5714285714287, 'NMHC <= 0.625\ngini = 0.265\nsamples = 500
\nvalue = [674, 126]\nclass = Yes'),
 Text(2580.75, 465.9428571428573, 'SO_2 <= 13.51\ngini = 0.174\nsamples = 459
\nvalue = [666, 71]\nclass = Yes'),
 Text(2545.875, 155.3142857142857, 'gini = 0.143\nsamples = 434\nvalue = [64
3, 54]\nclass = Yes'),
 Text(2615.625, 155.3142857142857, 'gini = 0.489\nsamples = 25\nvalue = [23,
17]\nclass = Yes'),
 Text(2720.25, 465.9428571428573, 'EBE <= 0.83\ngini = 0.222\nsamples = 41\nv
alue = [8, 55]\nclass = No'),
 Text(2685.375, 155.3142857142857, 'gini = 0.083\nsamples = 30\nvalue = [2, 4
4]\nclass = No'),
 Text(2755.125, 155.3142857142857, 'gini = 0.457\nsamples = 11\nvalue = [6, 1
1]\nclass = No'),
 Text(3069.0, 1087.2, 'NOx <= 45.155\ngini = 0.429\nsamples = 1652\nvalue =
[1772, 803]\nclass = Yes'),
 Text(2929.5, 776.5714285714287, 'NOx <= 31.29\ngini = 0.261\nsamples = 796\n
value = [1055, 193]\nclass = Yes'),
 Text(2859.75, 465.9428571428573, 'NO_2 <= 23.41\ngini = 0.13\nsamples = 369
\nvalue = [544, 41]\nclass = Yes'),
 Text(2824.875, 155.3142857142857, 'gini = 0.101\nsamples = 338\nvalue = [51
2, 29]\nclass = Yes'),
 Text(2894.625, 155.3142857142857, 'gini = 0.397\nsamples = 31\nvalue = [32,
```

```
12]\nclass = Yes'),
 Text(2999.25, 465.9428571428573, 'NO_2 <= 27.525\ngini = 0.353\nsamples = 42
7\nvalue = [511, 152]\nclass = Yes'),
 Text(2964.375, 155.3142857142857, 'gini = 0.193\nsamples = 164\nvalue = [22
2, 27]\nclass = Yes'),
 Text(3034.125, 155.3142857142857, 'gini = 0.422\nsamples = 263\nvalue = [28
9, 125]\nclass = Yes'),
 Text(3208.5, 776.5714285714287, 'CO <= 0.335\ngini = 0.497\nsamples = 856\nv
alue = [717, 610]\nclass = Yes'),
 Text(3138.75, 465.9428571428573, 'SO_2 <= 7.635\ngini = 0.472\nsamples = 559
\nvalue = [534, 330]\nclass = Yes'),
 Text(3103.875, 155.3142857142857, 'gini = 0.375\nsamples = 319\nvalue = [36
0, 120]\nclass = Yes'),
 Text(3173.625, 155.3142857142857, 'gini = 0.496\nsamples = 240\nvalue = [17
4, 210]\nclass = No'),
 Text(3278.25, 465.9428571428573, 'BEN <= 0.485\ngini = 0.478\nsamples = 297
\nvalue = [183, 280]\nclass = No'),
 Text(3243.375, 155.3142857142857, 'gini = 0.476\nsamples = 106\nvalue = [10
0, 64]\nclass = Yes'),
 Text(3313.125, 155.3142857142857, 'gini = 0.401\nsamples = 191\nvalue = [83,
216]\nclass = No'),
 Text(3906.0, 1397.8285714285716, 'NO_2 <= 39.96\ngini = 0.487\nsamples = 213
4\nvalue = [1960, 1418]\nclass = Yes'),
 Text(3627.0, 1087.2, 'CO <= 0.305\ngini = 0.239\nsamples = 559\nvalue = [77
0, 124]\nclass = Yes'),
 Text(3487.5, 776.5714285714287, 'O_3 <= 49.575\ngini = 0.134\nsamples = 299
\nvalue = [438, 34]\nclass = Yes'),
 Text(3417.75, 465.9428571428573, 'CO <= 0.245\ngini = 0.318\nsamples = 74\nv
alue = [89, 22]\nclass = Yes'),
 Text(3382.875, 155.3142857142857, 'gini = 0.191\nsamples = 37\nvalue = [50,
6]\nclass = Yes'),
 Text(3452.625, 155.3142857142857, 'gini = 0.413\nsamples = 37\nvalue = [39,
16]\nclass = Yes'),
 Text(3557.25, 465.9428571428573, 'station <= 28079062.0\ngini = 0.064\nsampl
es = 225\nvalue = [349, 12]\nclass = Yes'),
 Text(3522.375, 155.3142857142857, 'gini = 0.019\nsamples = 197\nvalue = [31
6, 3]\nclass = Yes'),
 Text(3592.125, 155.3142857142857, 'gini = 0.337\nsamples = 28\nvalue = [33,
9]\nclass = Yes'),
 Text(3766.5, 776.5714285714287, 'NOx <= 27.985\ngini = 0.336\nsamples = 260
\nvalue = [332, 90]\nclass = Yes'),
 Text(3696.75, 465.9428571428573, 'SO_2 <= 6.775\ngini = 0.019\nsamples = 73
\nvalue = [104, 1]\nclass = Yes'),
 Text(3661.875, 155.3142857142857, 'gini = 0.064\nsamples = 20\nvalue = [29,
1]\nclass = Yes'),
 Text(3731.625, 155.3142857142857, 'gini = 0.0\nsamples = 53\nvalue = [75, 0]
\nclass = Yes'),
 Text(3836.25, 465.9428571428573, 'SO_2 <= 6.305\ngini = 0.404\nsamples = 187
\nvalue = [228, 89]\nclass = Yes'),
 Text(3801.375, 155.3142857142857, 'gini = 0.351\nsamples = 12\nvalue = [5, 1
7]\nclass = No'),
 Text(3871.125, 155.3142857142857, 'gini = 0.369\nsamples = 175\nvalue = [22
3, 72]\nclass = Yes'),
 Text(4185.0, 1087.2, 'CO <= 0.355\ngini = 0.499\nsamples = 1575\nvalue = [11
90, 1294]\nclass = No'),
 Text(4045.5, 776.5714285714287, 'EBE <= 1.295\ngini = 0.391\nsamples = 487\n
value = [545, 198]\nclass = Yes'),
```

```
 Text(3975.75, 465.9428571428573, 'PM25 <= 17.23\ngini = 0.304\nsamples = 350
\nvalue = [439, 101]\nclass = Yes'),
 Text(3940.875, 155.3142857142857, 'gini = 0.256\nsamples = 303\nvalue = [39
9, 71]\nclass = Yes'),
 Text(4010.625, 155.3142857142857, 'gini = 0.49\nsamples = 47\nvalue = [40, 3
0]\nclass = Yes'),
 Text(4115.25, 465.9428571428573, 'PM25 <= 5.83\ngini = 0.499\nsamples = 137
\nvalue = [106, 97]\nclass = Yes'),
 Text(4080.375, 155.3142857142857, 'gini = 0.165\nsamples = 17\nvalue = [20,
2]\nclass = Yes'),
 Text(4150.125, 155.3142857142857, 'gini = 0.499\nsamples = 120\nvalue = [86,
95]\nclass = No'),
 Text(4324.5, 776.5714285714287, 'SO_2 <= 10.805\ngini = 0.466\nsamples = 108
8\nvalue = [645, 1096]\nclass = No'),
 Text(4254.75, 465.9428571428573, 'CO <= 0.605\ngini = 0.499\nsamples = 655\n
value = [497, 551]\nclass = No'),
 Text(4219.875, 155.3142857142857, 'gini = 0.497\nsamples = 522\nvalue = [45
0, 385]\nclass = Yes'),
 Text(4289.625, 155.3142857142857, 'gini = 0.344\nsamples = 133\nvalue = [47,
166]\nclass = No'),
 Text(4394.25, 465.9428571428573, 'NMHC <= 0.105\ngini = 0.336\nsamples = 433
\nvalue = [148, 545]\nclass = No'),
 Text(4359.375, 155.3142857142857, 'gini = 0.382\nsamples = 72\nvalue = [81,
28]\nclass = Yes'),
 Text(4429.125, 155.3142857142857, 'gini = 0.203\nsamples = 361\nvalue = [67,
517]\nclass = No')]
```



In [40]:
```
print("Linear:",lis)
print("Lasso:",las)
print("Ridge:",rrs)
print("ElasticNet:",ens)
print("Logistic:",los)
print("Random Forest:",rfcs)
```

```
Linear: 0.8783680625661898
Lasso: 0.37328599597971324
Ridge: 0.8783497693150732
ElasticNet: 0.575045121851463
Logistic: 0.5225188781014024
Random Forest: 0.8625513973793
```

Best Model is Linear Regression

2010

In [41]:
```
df2=pd.read_csv("madrid_2010.csv")
df2
```

Out[41]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	P
0	2010-03-01 01:00:00	NaN	0.29	NaN	NaN	NaN	25.090000	29.219999	NaN	68.930000	
1	2010-03-01 01:00:00	NaN	0.27	NaN	NaN	NaN	24.879999	30.040001	NaN	NaN	
2	2010-03-01 01:00:00	NaN	0.28	NaN	NaN	NaN	17.410000	20.540001	NaN	72.120003	
3	2010-03-01 01:00:00	0.38	0.24	1.74	NaN	0.05	15.610000	21.080000	NaN	72.970001	19.41(
4	2010-03-01 01:00:00	0.79	NaN	1.32	NaN	NaN	21.430000	26.070000	NaN	NaN	24.67(
...	...	...	...	...	...	...	...	...	...	...	
209443	2010-08-01 00:00:00	NaN	0.55	NaN	NaN	NaN	125.000000	219.899994	NaN	25.379999	
209444	2010-08-01 00:00:00	NaN	0.27	NaN	NaN	NaN	45.709999	47.410000	NaN	NaN	51.25(
209445	2010-08-01 00:00:00	NaN	NaN	NaN	NaN	0.24	46.560001	49.040001	NaN	46.250000	
209446	2010-08-01 00:00:00	NaN	NaN	NaN	NaN	NaN	46.770000	50.119999	NaN	77.709999	
209447	2010-08-01 00:00:00	0.92	0.43	0.71	NaN	0.25	76.330002	88.190002	NaN	52.259998	47.15(

209448 rows × 17 columns

In [42]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209448 entries, 0 to 209447
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     209448 non-null  object 
 1   BEN      60268 non-null   float64
 2   CO       94982 non-null   float64
 3   EBE      60253 non-null   float64
 4   MXY      6750 non-null    float64
 5   NMHC     51727 non-null   float64
 6   NO_2     208219 non-null  float64
 7   NOx      208210 non-null  float64
 8   OXY      6750 non-null    float64
 9   O_3      126684 non-null  float64
 10  PM10     106186 non-null  float64
 11  PM25     55514 non-null   float64
 12  PXY      6740 non-null    float64
 13  SO_2     93184 non-null   float64
 14  TCH      51730 non-null   float64
 15  TOL      60171 non-null   float64
 16  station  209448 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 27.2+ MB
```

In [43]:
```
df3=df2.dropna()
df3
```

Out[43]:

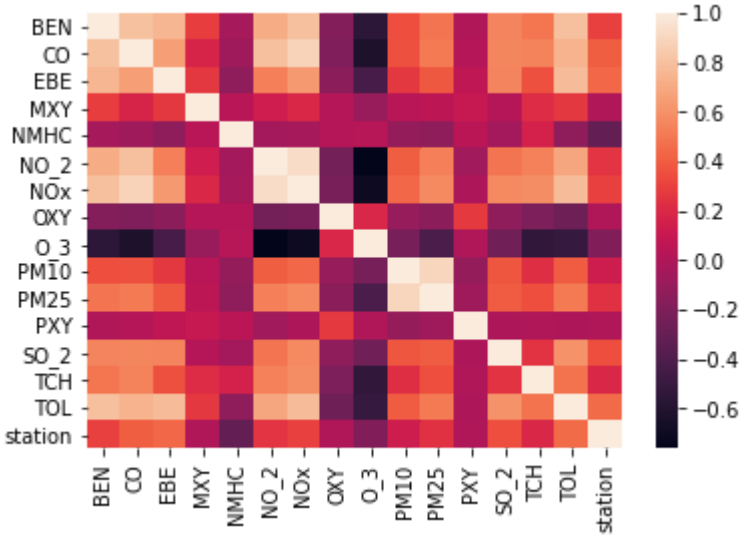
	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM1
11	2010-03-01 01:00:00	0.78	0.18	0.84	0.73	0.28	10.420000	11.900000	1.0	90.309998	18.3700
23	2010-03-01 01:00:00	0.70	0.23	1.00	0.73	0.18	17.820000	22.290001	1.0	70.550003	23.6399
35	2010-03-01 02:00:00	0.58	0.17	0.84	0.73	0.28	3.500000	4.950000	1.0	68.849998	5.6000
47	2010-03-01 02:00:00	0.33	0.21	0.84	0.73	0.17	10.810000	14.900000	1.0	74.750000	7.8900
59	2010-03-01 03:00:00	0.38	0.16	0.64	1.00	0.26	2.750000	4.200000	1.0	93.629997	5.1300
...	...	...	...	...	...	...	...	...	...	...	
191879	2010-05-31 22:00:00	0.60	0.26	0.82	0.13	0.16	33.360001	43.779999	1.0	38.459999	20.3400
191891	2010-05-31 23:00:00	0.41	0.16	0.71	0.19	0.10	24.299999	26.059999	1.0	50.290001	14.3800
191903	2010-05-31 23:00:00	0.57	0.28	0.64	0.19	0.18	35.540001	44.590000	1.0	34.020000	22.8400
191915	2010-06-01 00:00:00	0.34	0.16	0.69	0.22	0.10	23.559999	25.209999	1.0	45.930000	10.7700
191927	2010-06-01 00:00:00	0.43	0.25	0.79	0.22	0.18	34.910000	42.369999	1.0	29.540001	15.3500

6666 rows × 17 columns

In [44]:
```
df3=df3.drop(["date"],axis=1)
```

In [45]:
```
sns.heatmap(df3.corr())
```

Out[45]: <AxesSubplot:>



In [46]:
```
x=df3.drop(["TCH"],axis=1)
y=df3["TCH"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

Linear

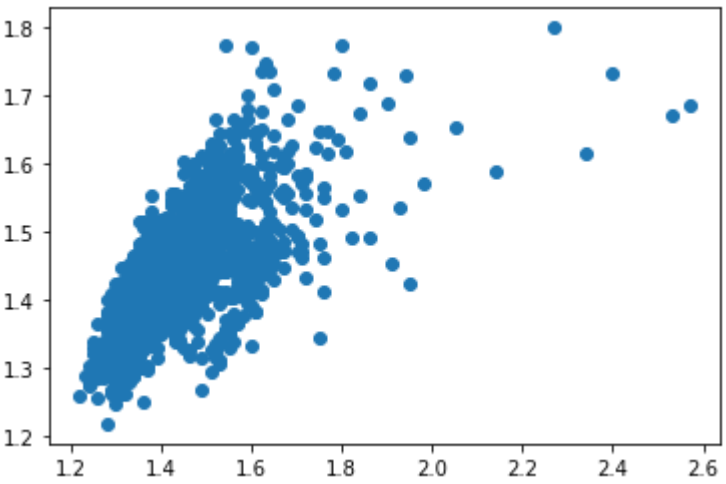
In [47]:
```
li=LinearRegression()
li.fit(x_train,y_train)
```

Out[47]: LinearRegression()

In [ ]:
```

```

In [48]:
```
prediction=li.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[48]: <matplotlib.collections.PathCollection at 0x23c8864f190>



In [49]:
```
lis=li.score(x_test,y_test)
```

In [50]:
```
df3["TCH"].value_counts()
```

Out[50]:
```
1.36    364
1.38    351
1.39    324
1.35    323
1.37    321
       ...
2.07      1
2.17      1
2.53      1
2.12      1
2.05      1
Name: TCH, Length: 100, dtype: int64
```

In [51]:
```
df3.loc[df3["TCH"]<1.40,"TCH"]=1
df3.loc[df3["TCH"]>1.40,"TCH"]=2
df3["TCH"].value_counts()
```

Out[51]:
```
1.0    3340
2.0    3326
Name: TCH, dtype: int64
```

In [ ]:
```

```

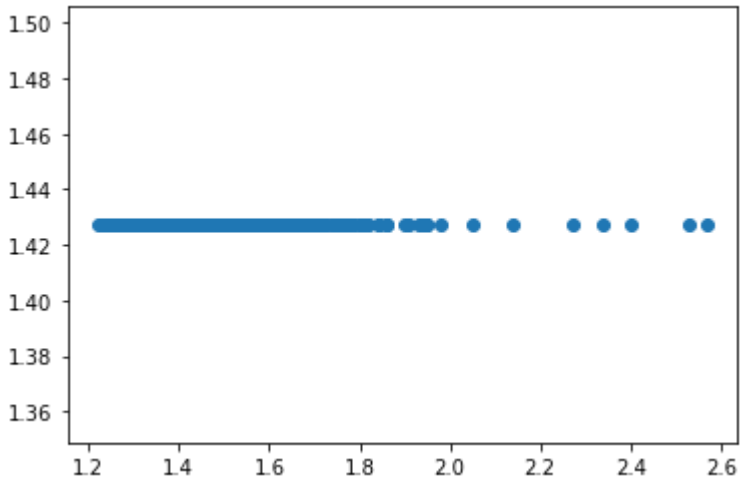
Lasso

In [52]:
```
la=Lasso(alpha=5)
la.fit(x_train,y_train)
```

Out[52]:
```
Lasso(alpha=5)
```

In [53]:
```
prediction1=la.predict(x_test)
plt.scatter(y_test,prediction1)
```

Out[53]: <matplotlib.collections.PathCollection at 0x23c886ac280>



In [54]:
```
las=la.score(x_test,y_test)
```

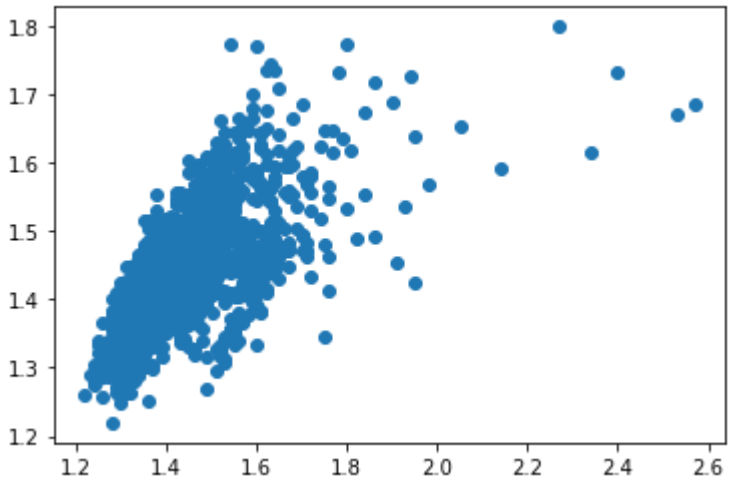
Ridge

In [55]:
```
rr=Ridge(alpha=1)
rr.fit(x_train,y_train)
```

Out[55]: Ridge(alpha=1)

In [56]:
```
prediction2=rr.predict(x_test)
plt.scatter(y_test,prediction2)
```

Out[56]: <matplotlib.collections.PathCollection at 0x23c8870a310>



In [57]:
```
rrs=rr.score(x_test,y_test)
```

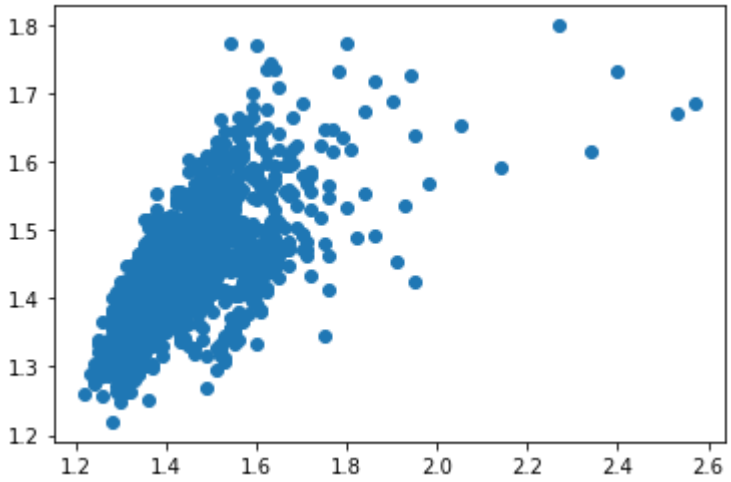
ElasticNet

In [58]:
```
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[58]:
```
ElasticNet()
```

In [59]:
```
prediction2=rr.predict(x_test)
plt.scatter(y_test,prediction2)
```

Out[59]:
```
<matplotlib.collections.PathCollection at 0x23c88762e20>
```



In [60]:
```
ens=en.score(x_test,y_test)
```

In [61]:
```
print(rr.score(x_test,y_test))
rr.score(x_train,y_train)
```

```
0.4590393479321705
```

Out[61]:
```
0.44781912704161386
```

Logistic

In [62]:
```
g={"TCH":{1.0:"Low",2.0:"High"}}
df3=df3.replace(g)
df3["TCH"].value_counts()
```

Out[62]:
```
Low     3340
High    3326
Name: TCH, dtype: int64
```

In [63]:
```
x=df3.drop(["TCH"],axis=1)
y=df3["TCH"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [64]:
```
lo=LogisticRegression()
lo.fit(x_train,y_train)
```

Out[64]:
```
LogisticRegression()
```

In [65]:
```
prediction3=lo.predict(x_test)
plt.scatter(y_test,prediction3)
```

Out[65]:
```
<matplotlib.collections.PathCollection at 0x23c8813aac0>
```

In [66]:
```
los=lo.score(x_test,y_test)
```

Random Forest

In [67]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
```

In [68]:
```
g1={"TCH":{"Low":1.0,"High":2.0}}
df3=df3.replace(g1)
```

In [69]:
```
x=df3.drop(["TCH"],axis=1)
y=df3["TCH"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [70]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[70]:
```
RandomForestClassifier()
```

In [71]:
```
parameter={
    'max_depth':[1,2,4,5,6],
    'min_samples_leaf':[5,10,15,20,25],
    'n_estimators':[10,20,30,40,50]
}
```

In [72]:
```
grid_search=GridSearchCV(estimator=rfc,param_grid=parameter,cv=2,scoring="accu
grid_search.fit(x_train,y_train)
```

Out[72]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 4, 5, 6],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

In [73]:
```
rfcs=grid_search.best_score_
```

In [74]:
```
rfc_best=grid_search.best_estimator_
```

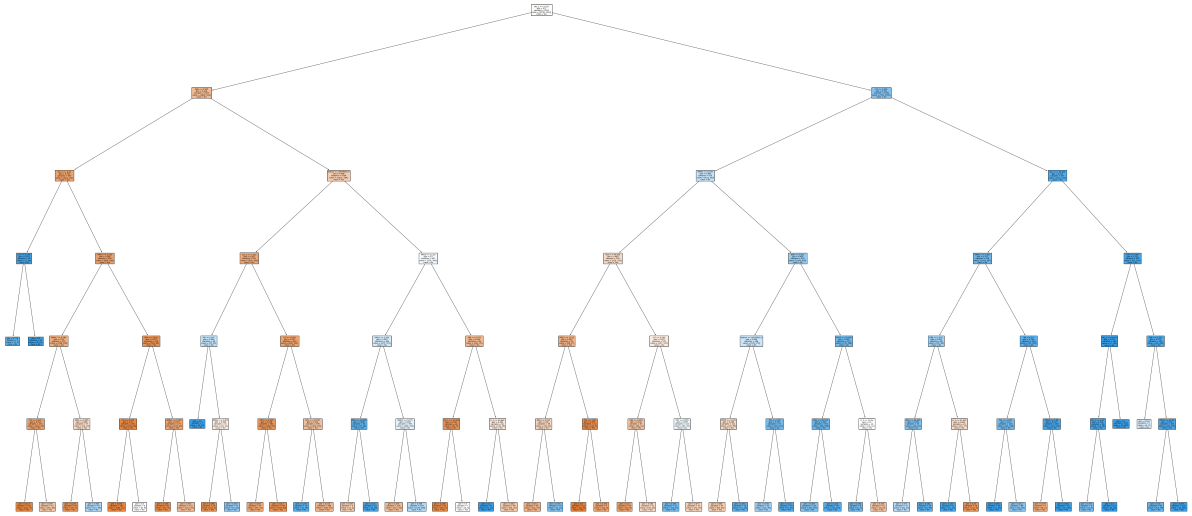
In [75]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes',"
```

Out[75]:
```
[Text(2028.8446601941748, 2019.0857142857144, 'NO_2 <= 32.59\ngini = 0.5\nsam
ples = 2951\nvalue = [2356, 2310]\nclass = Yes'),
 Text(753.0291262135922, 1708.457142857143, 'EBE <= 0.505\ngini = 0.428\nsamp
les = 1655\nvalue = [1802, 810]\nclass = Yes'),
 Text(238.36893203883497, 1397.8285714285716, 'SO_2 <= 6.42\ngini = 0.333\nsa
mples = 647\nvalue = [801, 214]\nclass = Yes'),
 Text(86.67961165048544, 1087.2, 'CO <= 0.175\ngini = 0.071\nsamples = 18\nva
lue = [1, 26]\nclass = No'),
 Text(43.33980582524272, 776.5714285714287, 'gini = 0.278\nsamples = 5\nvalue
= [1, 5]\nclass = No'),
 Text(130.01941747572818, 776.5714285714287, 'gini = 0.0\nsamples = 13\nvalue
= [0, 21]\nclass = No'),
 Text(390.05825242718447, 1087.2, 'PM10 <= 9.235\ngini = 0.308\nsamples = 629
\nvalue = [800, 188]\nclass = Yes'),
 Text(216.6990291262136, 776.5714285714287, 'NMHC <= 0.385\ngini = 0.39\nsamp
les = 251\nvalue = [298, 108]\nclass = Yes'),
 Text(130.01941747572818, 465.9428571428573, 'CO <= 0.215\ngini = 0.317\nsamp
les = 174\nvalue = [224, 55]\nclass = Yes'),
 Text(86.67961165048544, 155.3142857142857, 'gini = 0.113\nsamples = 101\nval
ue = [141, 9]\nclass = Yes'),
 Text(173.35922330097088, 155.3142857142857, 'gini = 0.459\nsamples = 73\nval
ue = [83, 46]\nclass = Yes'),
 Text(303.37864077669906, 465.9428571428573, 'BEN <= 0.265\ngini = 0.486\nsam
ples = 77\nvalue = [74, 53]\nclass = Yes'),
 Text(260.03883495145635, 155.3142857142857, 'gini = 0.226\nsamples = 34\nval
ue = [47, 7]\nclass = Yes'),
 Text(346.71844660194176, 155.3142857142857, 'gini = 0.466\nsamples = 43\nval
ue = [27, 46]\nclass = No'),
 Text(563.4174757281554, 776.5714285714287, 'NOx <= 19.02\ngini = 0.237\nsamp
les = 378\nvalue = [502, 80]\nclass = Yes'),
 Text(476.73786407766994, 465.9428571428573, 'CO <= 0.225\ngini = 0.11\nsampl
es = 201\nvalue = [291, 18]\nclass = Yes'),
 Text(433.3980582524272, 155.3142857142857, 'gini = 0.078\nsamples = 192\nval
ue = [285, 12]\nclass = Yes'),
 Text(520.0776699029127, 155.3142857142857, 'gini = 0.5\nsamples = 9\nvalue =
[6, 6]\nclass = Yes'),
 Text(650.0970873786408, 465.9428571428573, 'NMHC <= 0.185\ngini = 0.351\nsam
ples = 177\nvalue = [211, 62]\nclass = Yes'),
 Text(606.7572815533981, 155.3142857142857, 'gini = 0.155\nsamples = 65\nvalu
e = [97, 9]\nclass = Yes'),
 Text(693.4368932038835, 155.3142857142857, 'gini = 0.433\nsamples = 112\nval
ue = [114, 53]\nclass = Yes'),
 Text(1267.6893203883496, 1397.8285714285716, 'station <= 28079062.0\ngini =
0.468\nsamples = 1008\nvalue = [1001, 596]\nclass = Yes'),
 Text(931.8058252427185, 1087.2, 'SO_2 <= 7.225\ngini = 0.367\nsamples = 521
\nvalue = [625, 200]\nclass = Yes'),
 Text(780.1165048543689, 776.5714285714287, 'PXY <= 0.47\ngini = 0.484\nsampl
es = 38\nvalue = [25, 36]\nclass = No'),
 Text(736.7766990291262, 465.9428571428573, 'gini = 0.0\nsamples = 7\nvalue =
[0, 14]\nclass = No'),
 Text(823.4563106796116, 465.9428571428573, 'NMHC <= 0.305\ngini = 0.498\nsam
ples = 31\nvalue = [25, 22]\nclass = Yes'),
 Text(780.1165048543689, 155.3142857142857, 'gini = 0.198\nsamples = 14\nvalu
e = [16, 2]\nclass = Yes'),
 Text(866.7961165048544, 155.3142857142857, 'gini = 0.428\nsamples = 17\nvalu
e = [9, 20]\nclass = No'),
 Text(1083.495145631068, 776.5714285714287, 'CO <= 0.195\ngini = 0.337\nsampl
```

```
es = 483\nvalue = [600, 164]\nclass = Yes'),
 Text(996.8155339805826, 465.9428571428573, 'MXY <= 0.325\ngini = 0.238\nsamp
les = 284\nvalue = [400, 64]\nclass = Yes'),
 Text(953.4757281553399, 155.3142857142857, 'gini = 0.354\nsamples = 37\nvalu
e = [47, 14]\nclass = Yes'),
 Text(1040.1553398058254, 155.3142857142857, 'gini = 0.217\nsamples = 247\nva
lue = [353, 50]\nclass = Yes'),
 Text(1170.1747572815534, 465.9428571428573, 'NOx <= 10.055\ngini = 0.444\nsa
mples = 199\nvalue = [200, 100]\nclass = Yes'),
 Text(1126.8349514563108, 155.3142857142857, 'gini = 0.245\nsamples = 14\nval
ue = [3, 18]\nclass = No'),
 Text(1213.5145631067962, 155.3142857142857, 'gini = 0.415\nsamples = 185\nva
lue = [197, 82]\nclass = Yes'),
 Text(1603.5728155339807, 1087.2, 'PM25 <= 11.25\ngini = 0.5\nsamples = 487\n
value = [376, 396]\nclass = No'),
 Text(1430.2135922330099, 776.5714285714287, 'PM10 <= 5.105\ngini = 0.493\nsa
mples = 389\nvalue = [275, 351]\nclass = No'),
 Text(1343.5339805825242, 465.9428571428573, 'NOx <= 14.835\ngini = 0.266\nsa
mples = 30\nvalue = [9, 48]\nclass = No'),
 Text(1300.1941747572816, 155.3142857142857, 'gini = 0.469\nsamples = 5\nvalu
e = [5, 3]\nclass = Yes'),
 Text(1386.873786407767, 155.3142857142857, 'gini = 0.15\nsamples = 25\nvalue
= [4, 45]\nclass = No'),
 Text(1516.8932038834953, 465.9428571428573, 'NMHC <= 0.175\ngini = 0.498\nsa
mples = 359\nvalue = [266, 303]\nclass = No'),
 Text(1473.5533980582525, 155.3142857142857, 'gini = 0.435\nsamples = 105\nva
lue = [117, 55]\nclass = Yes'),
 Text(1560.2330097087379, 155.3142857142857, 'gini = 0.469\nsamples = 254\nva
lue = [149, 248]\nclass = No'),
 Text(1776.9320388349515, 776.5714285714287, 'EBE <= 0.775\ngini = 0.426\nsam
ples = 98\nvalue = [101, 45]\nclass = Yes'),
 Text(1690.2524271844661, 465.9428571428573, 'SO_2 <= 10.265\ngini = 0.236\ns
amples = 42\nvalue = [57, 9]\nclass = Yes'),
 Text(1646.9126213592233, 155.3142857142857, 'gini = 0.158\nsamples = 35\nval
ue = [53, 5]\nclass = Yes'),
 Text(1733.5922330097087, 155.3142857142857, 'gini = 0.5\nsamples = 7\nvalue
= [4, 4]\nclass = Yes'),
 Text(1863.611650485437, 465.9428571428573, 'O_3 <= 49.865\ngini = 0.495\nsam
ples = 56\nvalue = [44, 36]\nclass = Yes'),
 Text(1820.2718446601943, 155.3142857142857, 'gini = 0.0\nsamples = 8\nvalue
= [0, 11]\nclass = No'),
 Text(1906.9514563106798, 155.3142857142857, 'gini = 0.462\nsamples = 48\nval
ue = [44, 25]\nclass = Yes'),
 Text(3304.6601941747576, 1708.457142857143, 'CO <= 0.345\ngini = 0.394\nsamp
les = 1296\nvalue = [554, 1500]\nclass = No'),
 Text(2643.728155339806, 1397.8285714285716, 'NMHC <= 0.195\ngini = 0.486\nsa
mples = 619\nvalue = [419, 590]\nclass = No'),
 Text(2297.009708737864, 1087.2, 'NOx <= 44.935\ngini = 0.485\nsamples = 226
\nvalue = [215, 152]\nclass = Yes'),
 Text(2123.650485436893, 776.5714285714287, 'PM25 <= 11.55\ngini = 0.407\nsam
ples = 59\nvalue = [68, 27]\nclass = Yes'),
 Text(2036.9708737864078, 465.9428571428573, 'BEN <= 0.615\ngini = 0.48\nsamp
les = 38\nvalue = [36, 24]\nclass = Yes'),
 Text(1993.6310679611652, 155.3142857142857, 'gini = 0.411\nsamples = 28\nval
ue = [32, 13]\nclass = Yes'),
 Text(2080.310679611651, 155.3142857142857, 'gini = 0.391\nsamples = 10\nvalu
e = [4, 11]\nclass = No'),
```

```
 Text(2210.330097087379, 465.9428571428573, 'TOL <= 1.49\ngini = 0.157\nsampl
es = 21\nvalue = [32, 3]\nclass = Yes'),
 Text(2166.990291262136, 155.3142857142857, 'gini = 0.0\nsamples = 12\nvalue
= [22, 0]\nclass = Yes'),
 Text(2253.6699029126216, 155.3142857142857, 'gini = 0.355\nsamples = 9\nvalu
e = [10, 3]\nclass = Yes'),
 Text(2470.3689320388353, 776.5714285714287, 'CO <= 0.295\ngini = 0.497\nsamp
les = 167\nvalue = [147, 125]\nclass = Yes'),
 Text(2383.6893203883496, 465.9428571428573, 'EBE <= 0.495\ngini = 0.449\nsam
ples = 57\nvalue = [60, 31]\nclass = Yes'),
 Text(2340.349514563107, 155.3142857142857, 'gini = 0.32\nsamples = 16\nvalue
= [24, 6]\nclass = Yes'),
 Text(2427.0291262135925, 155.3142857142857, 'gini = 0.484\nsamples = 41\nval
ue = [36, 25]\nclass = Yes'),
 Text(2557.0485436893205, 465.9428571428573, 'PXY <= 0.565\ngini = 0.499\nsam
ples = 110\nvalue = [87, 94]\nclass = No'),
 Text(2513.7087378640776, 155.3142857142857, 'gini = 0.354\nsamples = 37\nval
ue = [14, 47]\nclass = No'),
 Text(2600.3883495145633, 155.3142857142857, 'gini = 0.477\nsamples = 73\nval
ue = [73, 47]\nclass = Yes'),
 Text(2990.4466019417478, 1087.2, 'BEN <= 0.615\ngini = 0.434\nsamples = 393
\nvalue = [204, 438]\nclass = No'),
 Text(2817.087378640777, 776.5714285714287, 'station <= 28079062.0\ngini = 0.
489\nsamples = 186\nvalue = [126, 170]\nclass = No'),
 Text(2730.4077669902913, 465.9428571428573, 'PM25 <= 11.65\ngini = 0.486\nsa
mples = 111\nvalue = [97, 69]\nclass = Yes'),
 Text(2687.0679611650485, 155.3142857142857, 'gini = 0.466\nsamples = 97\nval
ue = [94, 55]\nclass = Yes'),
 Text(2773.747572815534, 155.3142857142857, 'gini = 0.291\nsamples = 14\nvalu
e = [3, 14]\nclass = No'),
 Text(2903.766990291262, 465.9428571428573, 'TOL <= 2.09\ngini = 0.347\nsampl
es = 75\nvalue = [29, 101]\nclass = No'),
 Text(2860.4271844660198, 155.3142857142857, 'gini = 0.448\nsamples = 35\nval
ue = [20, 39]\nclass = No'),
 Text(2947.106796116505, 155.3142857142857, 'gini = 0.221\nsamples = 40\nvalu
e = [9, 62]\nclass = No'),
 Text(3163.8058252427186, 776.5714285714287, 'PM25 <= 16.365\ngini = 0.349\ns
amples = 207\nvalue = [78, 268]\nclass = No'),
 Text(3077.126213592233, 465.9428571428573, 'NOx <= 59.515\ngini = 0.326\nsam
ples = 191\nvalue = [66, 256]\nclass = No'),
 Text(3033.7864077669906, 155.3142857142857, 'gini = 0.385\nsamples = 113\nva
lue = [50, 142]\nclass = No'),
 Text(3120.4660194174758, 155.3142857142857, 'gini = 0.216\nsamples = 78\nval
ue = [16, 114]\nclass = No'),
 Text(3250.485436893204, 465.9428571428573, 'O_3 <= 33.54\ngini = 0.5\nsample
s = 16\nvalue = [12, 12]\nclass = Yes'),
 Text(3207.1456310679614, 155.3142857142857, 'gini = 0.346\nsamples = 6\nvalu
e = [2, 7]\nclass = No'),
 Text(3293.8252427184466, 155.3142857142857, 'gini = 0.444\nsamples = 10\nval
ue = [10, 5]\nclass = Yes'),
 Text(3965.5922330097087, 1397.8285714285716, 'NO_2 <= 58.595\ngini = 0.225\n
samples = 677\nvalue = [135, 910]\nclass = No'),
 Text(3683.883495145631, 1087.2, 'NMHC <= 0.185\ngini = 0.327\nsamples = 318
\nvalue = [103, 397]\nclass = No'),
 Text(3510.5242718446602, 776.5714285714287, 'EBE <= 1.375\ngini = 0.453\nsam
ples = 98\nvalue = [52, 98]\nclass = No'),
 Text(3423.844660194175, 465.9428571428573, 'NOx <= 64.465\ngini = 0.388\nsam
```

```
ples = 74\nvalue = [29, 81]\nclass = No'),
 Text(3380.5048543689322, 155.3142857142857, 'gini = 0.473\nsamples = 37\nval
ue = [20, 32]\nclass = No'),
 Text(3467.1844660194174, 155.3142857142857, 'gini = 0.262\nsamples = 37\nval
ue = [9, 49]\nclass = No'),
 Text(3597.203883495146, 465.9428571428573, 'O_3 <= 21.545\ngini = 0.489\nsam
ples = 24\nvalue = [23, 17]\nclass = Yes'),
 Text(3553.864077669903, 155.3142857142857, 'gini = 0.133\nsamples = 8\nvalue
= [1, 13]\nclass = No'),
 Text(3640.5436893203887, 155.3142857142857, 'gini = 0.26\nsamples = 16\nvalu
e = [22, 4]\nclass = Yes'),
 Text(3857.2427184466023, 776.5714285714287, 'TOL <= 3.5\ngini = 0.249\nsampl
es = 220\nvalue = [51, 299]\nclass = No'),
 Text(3770.5631067961167, 465.9428571428573, 'TOL <= 2.165\ngini = 0.34\nsamp
les = 120\nvalue = [41, 148]\nclass = No'),
 Text(3727.223300970874, 155.3142857142857, 'gini = 0.038\nsamples = 34\nvalu
e = [1, 50]\nclass = No'),
 Text(3813.9029126213595, 155.3142857142857, 'gini = 0.412\nsamples = 86\nval
ue = [40, 98]\nclass = No'),
 Text(3943.9223300970875, 465.9428571428573, 'PM25 <= 5.56\ngini = 0.117\nsam
ples = 100\nvalue = [10, 151]\nclass = No'),
 Text(3900.5825242718447, 155.3142857142857, 'gini = 0.375\nsamples = 5\nvalu
e = [6, 2]\nclass = Yes'),
 Text(3987.2621359223303, 155.3142857142857, 'gini = 0.051\nsamples = 95\nval
ue = [4, 149]\nclass = No'),
 Text(4247.300970873786, 1087.2, 'OXY <= 0.99\ngini = 0.111\nsamples = 359\nv
alue = [32, 513]\nclass = No'),
 Text(4160.621359223302, 776.5714285714287, 'NO_2 <= 63.34\ngini = 0.017\nsam
ples = 149\nvalue = [2, 233]\nclass = No'),
 Text(4117.281553398058, 465.9428571428573, 'SO_2 <= 9.97\ngini = 0.124\nsamp
les = 20\nvalue = [2, 28]\nclass = No'),
 Text(4073.9417475728155, 155.3142857142857, 'gini = 0.375\nsamples = 5\nvalu
e = [2, 6]\nclass = No'),
 Text(4160.621359223302, 155.3142857142857, 'gini = 0.0\nsamples = 15\nvalue
= [0, 22]\nclass = No'),
 Text(4203.961165048544, 465.9428571428573, 'gini = 0.0\nsamples = 129\nvalue
= [0, 205]\nclass = No'),
 Text(4333.980582524272, 776.5714285714287, 'EBE <= 0.375\ngini = 0.175\nsamp
les = 210\nvalue = [30, 280]\nclass = No'),
 Text(4290.64077669903, 465.9428571428573, 'gini = 0.494\nsamples = 5\nvalue
= [4, 5]\nclass = No'),
 Text(4377.320388349514, 465.9428571428573, 'PM10 <= 20.64\ngini = 0.158\nsam
ples = 205\nvalue = [26, 275]\nclass = No'),
 Text(4333.980582524272, 155.3142857142857, 'gini = 0.31\nsamples = 69\nvalue
= [18, 76]\nclass = No'),
 Text(4420.660194174758, 155.3142857142857, 'gini = 0.074\nsamples = 136\nval
ue = [8, 199]\nclass = No')]
```



In [76]:
```
print("Linear:",lis)
print("Lasso:",las)
print("Ridge:",rrs)
print("ElasticNet:",ens)
print("Logistic:",los)
print("Random Forest:",rfcs)
```

```
Linear: 0.45865884208093644
Lasso: -0.0002593585325185721
Ridge: 0.4590393479321705
ElasticNet: 0.35747558851830485
Logistic: 0.4895
Random Forest: 0.7788255465066438
```

Best model is Random Forest