In [478]:

```
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 [608]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\csvs_per_year\csvs_per_year\madrid_201
a
```

Out[608]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2013-11-01 01:00:00	NaN	0.6	NaN	NaN	135.0	74.0	NaN	NaN	NaN	7.0	NaN	NaN	28079004
1	2013-11-01 01:00:00	1.5	0.5	1.3	NaN	71.0	83.0	2.0	23.0	16.0	12.0	NaN	8.3	28079008
2	2013-11-01 01:00:00	3.9	NaN	2.8	NaN	49.0	70.0	NaN	NaN	NaN	NaN	NaN	9.0	28079011
3	2013-11-01 01:00:00	NaN	0.5	NaN	NaN	82.0	87.0	3.0	NaN	NaN	NaN	NaN	NaN	28079016
4	2013-11-01 01:00:00	NaN	NaN	NaN	NaN	242.0	111.0	2.0	NaN	NaN	12.0	NaN	NaN	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
209875	2013-03-01 00:00:00	NaN	0.4	NaN	NaN	8.0	39.0	52.0	NaN	NaN	NaN	NaN	NaN	28079056
209876	2013-03-01 00:00:00	NaN	0.4	NaN	NaN	1.0	11.0	NaN	6.0	NaN	2.0	NaN	NaN	28079057
209877	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	2.0	4.0	75.0	NaN	NaN	NaN	NaN	NaN	28079058
209878	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	2.0	11.0	52.0	NaN	NaN	NaN	NaN	NaN	28079059
209879	2013-03-01 00:00:00	NaN	NaN	NaN	NaN	1.0	10.0	75.0	3.0	NaN	NaN	NaN	NaN	28079060

209880 rows × 14 columns

In [609]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 209880 entries, 0 to 209879
Data columns (total 14 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     209880 non-null  object 
 1   BEN      50462 non-null   float64
 2   CO       87018 non-null   float64
 3   EBE      50463 non-null   float64
 4   NMHC     25935 non-null   float64
 5   NO       209108 non-null  float64
 6   NO_2     209108 non-null  float64
 7   O_3      121858 non-null  float64
 8   PM10     104339 non-null  float64
 9   PM25     51980 non-null   float64
 10  SO_2     86970 non-null   float64
 11  TCH      25935 non-null   float64
 12  TOL      50317 non-null   float64
 13  station  209880 non-null  int64  
dtypes: float64(12), int64(1), object(1)
memory usage: 22.4+ MB
```

In [610]:

```
b=a.fillna(value=55)
b
```

Out[610]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2013-11-01 01:00:00	55.0	0.6	55.0	55.0	135.0	74.0	55.0	55.0	55.0	7.0	55.0	55.0	28079004
1	2013-11-01 01:00:00	1.5	0.5	1.3	55.0	71.0	83.0	2.0	23.0	16.0	12.0	55.0	8.3	28079008
2	2013-11-01 01:00:00	3.9	55.0	2.8	55.0	49.0	70.0	55.0	55.0	55.0	55.0	55.0	9.0	28079011
3	2013-11-01 01:00:00	55.0	0.5	55.0	55.0	82.0	87.0	3.0	55.0	55.0	55.0	55.0	55.0	28079016
4	2013-11-01 01:00:00	55.0	55.0	55.0	55.0	242.0	111.0	2.0	55.0	55.0	12.0	55.0	55.0	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
209875	2013-03-01 00:00:00	55.0	0.4	55.0	55.0	8.0	39.0	52.0	55.0	55.0	55.0	55.0	55.0	28079056
209876	2013-03-01 00:00:00	55.0	0.4	55.0	55.0	1.0	11.0	55.0	6.0	55.0	2.0	55.0	55.0	28079057
209877	2013-03-01 00:00:00	55.0	55.0	55.0	55.0	2.0	4.0	75.0	55.0	55.0	55.0	55.0	55.0	28079058
209878	2013-03-01 00:00:00	55.0	55.0	55.0	55.0	2.0	11.0	52.0	55.0	55.0	55.0	55.0	55.0	28079059
209879	2013-03-01 00:00:00	55.0	55.0	55.0	55.0	1.0	10.0	75.0	3.0	55.0	55.0	55.0	55.0	28079060

209880 rows × 14 columns

In [611]:

```
b.columns
```

Out[611]:

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

In [612]:

```
c=b.head(20)
c
```

Out[612]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2013-11-01 01:00:00	55.0	0.6	55.0	55.00	135.0	74.0	55.0	55.0	55.0	7.0	55.00	55.0	28079004
1	2013-11-01 01:00:00	1.5	0.5	1.3	55.00	71.0	83.0	2.0	23.0	16.0	12.0	55.00	8.3	28079008
2	2013-11-01 01:00:00	3.9	55.0	2.8	55.00	49.0	70.0	55.0	55.0	55.0	55.0	55.00	9.0	28079011
3	2013-11-01 01:00:00	55.0	0.5	55.0	55.00	82.0	87.0	3.0	55.0	55.0	55.0	55.00	55.0	28079016
4	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	242.0	111.0	2.0	55.0	55.0	12.0	55.00	55.0	28079017
5	2013-11-01 01:00:00	1.0	0.6	0.8	55.00	70.0	70.0	2.0	24.0	55.0	6.0	55.00	5.2	28079018
6	2013-11-01 01:00:00	55.0	0.4	55.0	0.29	51.0	80.0	5.0	23.0	14.0	4.0	1.44	55.0	28079024
7	2013-11-01 01:00:00	55.0	55.0	55.0	0.23	29.0	60.0	4.0	55.0	55.0	55.0	1.51	55.0	28079027
8	2013-11-01 01:00:00	55.0	1.0	55.0	55.00	165.0	107.0	2.0	55.0	55.0	11.0	55.00	55.0	28079035
9	2013-11-01 01:00:00	55.0	0.6	55.0	55.00	63.0	93.0	55.0	11.0	55.0	8.0	55.00	55.0	28079036
10	2013-11-01 01:00:00	1.4	55.0	1.4	55.00	68.0	84.0	55.0	26.0	11.0	6.0	55.00	7.4	28079038
11	2013-11-01 01:00:00	55.0	0.6	55.0	55.00	60.0	82.0	5.0	55.0	55.0	55.0	55.00	55.0	28079039
12	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	69.0	96.0	55.0	29.0	55.0	5.0	55.00	55.0	28079040
13	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	122.0	72.0	55.0	32.0	20.0	55.0	55.00	55.0	28079047
14	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	43.0	81.0	55.0	13.0	10.0	55.0	55.00	55.0	28079048
15	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	132.0	77.0	2.0	55.0	55.0	55.0	55.00	55.0	28079049
16	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	102.0	73.0	55.0	19.0	9.0	55.0	55.00	55.0	28079050
17	2013-11-01 01:00:00	55.0	55.0	55.0	55.00	169.0	74.0	3.0	55.0	55.0	55.0	55.00	55.0	28079054
18	2013-11-01 01:00:00	1.6	55.0	1.4	0.22	62.0	72.0	55.0	20.0	55.0	55.0	1.67	5.4	28079055
19	2013-11-01 01:00:00	55.0	0.8	55.0	55.00	115.0	87.0	5.0	55.0	55.0	55.0	55.00	55.0	28079056

In [613]:

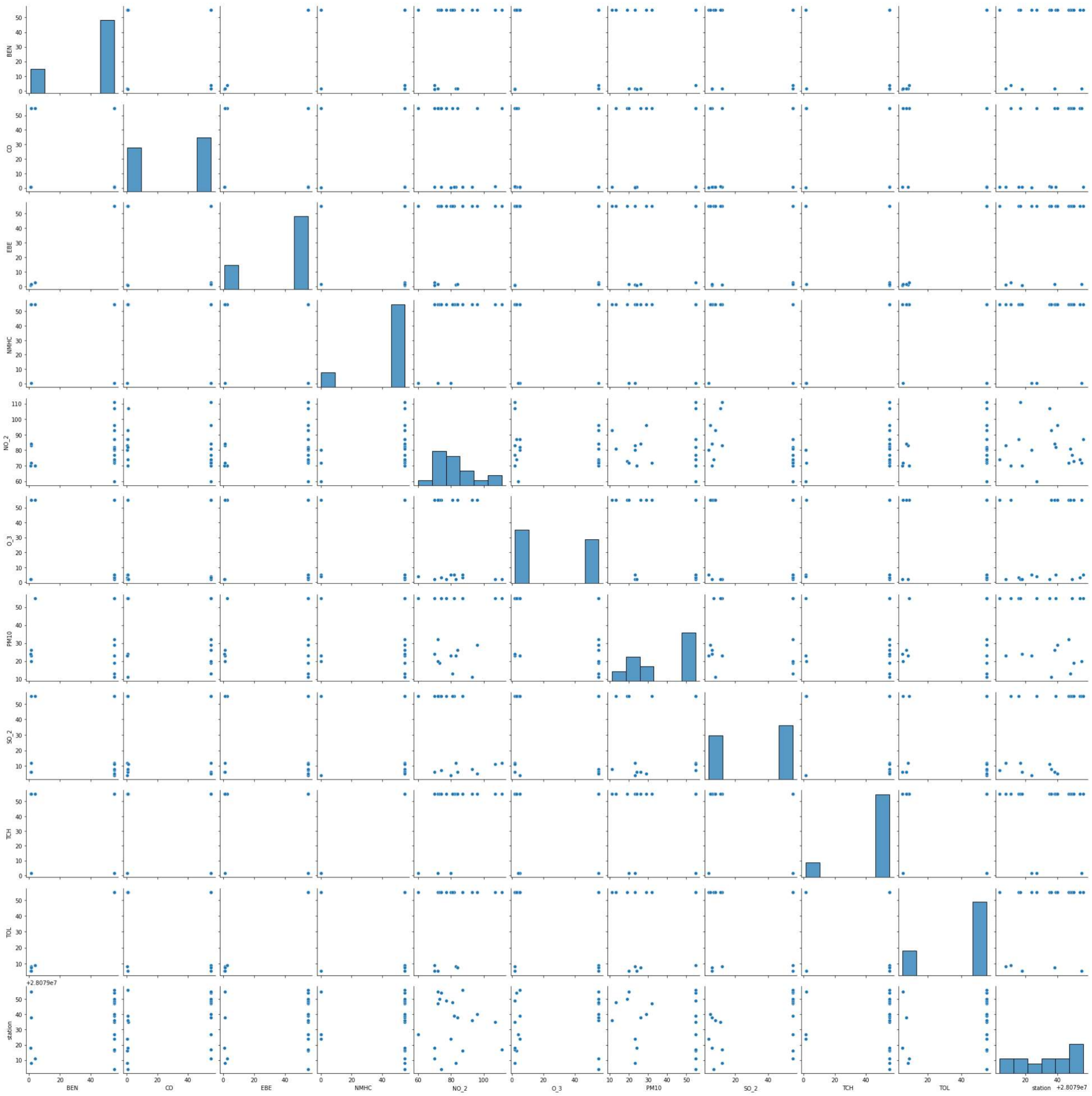
```
d=c[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
d
```

Out[613]:

	BEN	CO	EBE	NMHC	NO_2	O_3	PM10	SO_2	TCH	TOL	station
0	55.0	0.6	55.0	55.00	74.0	55.0	55.0	7.0	55.00	55.0	28079004
1	1.5	0.5	1.3	55.00	83.0	2.0	23.0	12.0	55.00	8.3	28079008
2	3.9	55.0	2.8	55.00	70.0	55.0	55.0	55.0	55.00	9.0	28079011
3	55.0	0.5	55.0	55.00	87.0	3.0	55.0	55.0	55.00	55.0	28079016
4	55.0	55.0	55.0	55.00	111.0	2.0	55.0	12.0	55.00	55.0	28079017
5	1.0	0.6	0.8	55.00	70.0	2.0	24.0	6.0	55.00	5.2	28079018
6	55.0	0.4	55.0	0.29	80.0	5.0	23.0	4.0	1.44	55.0	28079024
7	55.0	55.0	55.0	0.23	60.0	4.0	55.0	55.0	1.51	55.0	28079027
8	55.0	1.0	55.0	55.00	107.0	2.0	55.0	11.0	55.00	55.0	28079035
9	55.0	0.6	55.0	55.00	93.0	55.0	11.0	8.0	55.00	55.0	28079036
10	1.4	55.0	1.4	55.00	84.0	55.0	26.0	6.0	55.00	7.4	28079038
11	55.0	0.6	55.0	55.00	82.0	5.0	55.0	55.0	55.00	55.0	28079039
12	55.0	55.0	55.0	55.00	96.0	55.0	29.0	5.0	55.00	55.0	28079040
13	55.0	55.0	55.0	55.00	72.0	55.0	32.0	55.0	55.00	55.0	28079047
14	55.0	55.0	55.0	55.00	81.0	55.0	13.0	55.0	55.00	55.0	28079048
15	55.0	55.0	55.0	55.00	77.0	2.0	55.0	55.0	55.00	55.0	28079049
16	55.0	55.0	55.0	55.00	73.0	55.0	19.0	55.0	55.00	55.0	28079050
17	55.0	55.0	55.0	55.00	74.0	3.0	55.0	55.0	55.00	55.0	28079054
18	1.6	55.0	1.4	0.22	72.0	55.0	20.0	55.0	1.67	5.4	28079055
19	55.0	0.8	55.0	55.00	87.0	5.0	55.0	55.0	55.00	55.0	28079056

In [614]:
```
sns.pairplot(d)
```

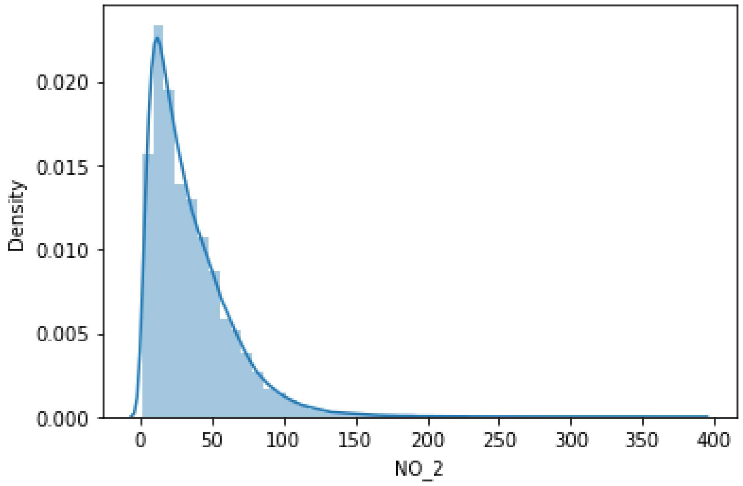
Out[614]:
```
<seaborn.axisgrid.PairGrid at 0x1b6f8031670>
```



In [615]:
```
sns.distplot(a['NO_2'])
```

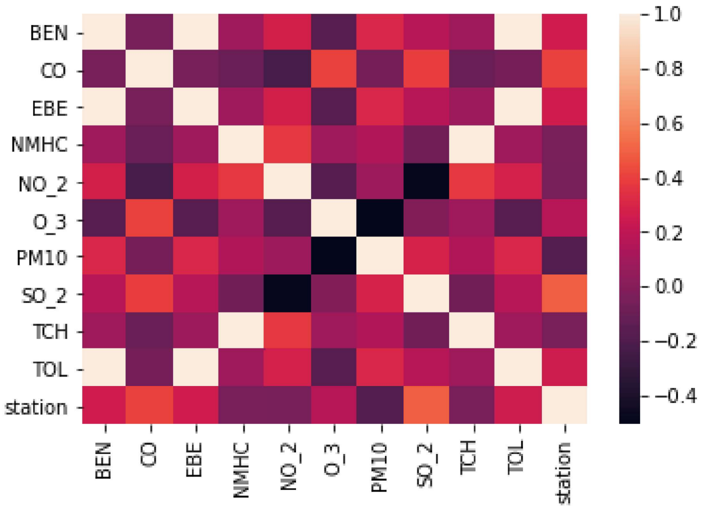
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[615]:
```
<AxesSubplot:xlabel='NO_2', ylabel='Density'>
```



In [616]:
```
sns.heatmap(d.corr())
```

Out[616]:
```
<AxesSubplot:>
```



In [617]:
```
x=d[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2']]
y=d['TCH']
```

In [618]:
```
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 [619]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[619]:
```
LinearRegression()
```

In [620]:
```
print(lr.intercept_)
```

```
-7.686001097830285
```

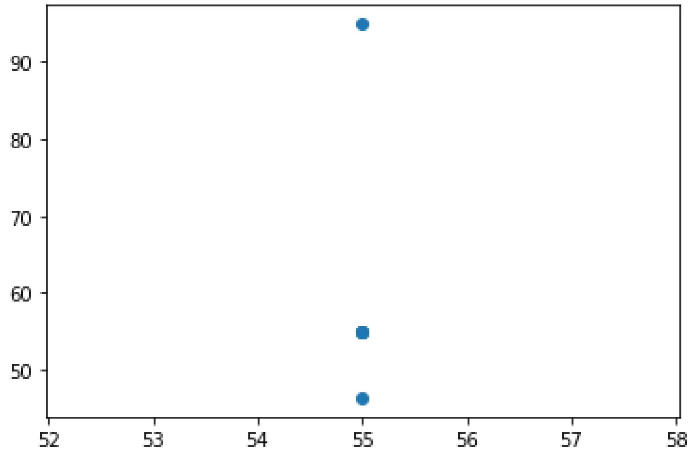
In [621]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[621]:

	Co-efficient
BEN	43.983082
CO	0.000888
EBE	-43.821223
NMHC	0.977052
NO_2	0.000138

In [622]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[622]:
```
<matplotlib.collections.PathCollection at 0x1b71ec21bb0>
```



In [623]:
```
print(lr.score(x_test,y_test))
```

```
0.0
```

In [624]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [625]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[625]:
```
Ridge(alpha=10)
```

In [626]:
```
rr.score(x_test,y_test)
```

Out[626]:
```
0.0
```

In [627]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[627]:
```
Lasso(alpha=10)
```

In [628]:
```
la.score(x_test,y_test)
```

Out[628]:
```
0.0
```

In [629]:
```
a1=b.head(7000)
a1
```

Out[629]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	station
0	2013-11-01 01:00:00	55.0	0.6	55.0	55.0	135.0	74.0	55.0	55.0	55.0	7.0	55.0	55.0	28079004
1	2013-11-01 01:00:00	1.5	0.5	1.3	55.0	71.0	83.0	2.0	23.0	16.0	12.0	55.0	8.3	28079008
2	2013-11-01 01:00:00	3.9	55.0	2.8	55.0	49.0	70.0	55.0	55.0	55.0	55.0	55.0	9.0	28079011
3	2013-11-01 01:00:00	55.0	0.5	55.0	55.0	82.0	87.0	3.0	55.0	55.0	55.0	55.0	55.0	28079016
4	2013-11-01 01:00:00	55.0	55.0	55.0	55.0	242.0	111.0	2.0	55.0	55.0	12.0	55.0	55.0	28079017
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
6995	2013-11-13 04:00:00	55.0	0.2	55.0	55.0	1.0	8.0	40.0	55.0	55.0	55.0	55.0	55.0	28079039
6996	2013-11-13 04:00:00	55.0	55.0	55.0	55.0	1.0	5.0	55.0	3.0	55.0	1.0	55.0	55.0	28079040
6997	2013-11-13 04:00:00	55.0	55.0	55.0	55.0	1.0	6.0	55.0	3.0	2.0	55.0	55.0	55.0	28079047
6998	2013-11-13 04:00:00	55.0	55.0	55.0	55.0	1.0	9.0	55.0	5.0	1.0	55.0	55.0	55.0	28079048
6999	2013-11-13 04:00:00	55.0	55.0	55.0	55.0	1.0	9.0	43.0	55.0	55.0	55.0	55.0	55.0	28079049

7000 rows × 14 columns

In [630]:
```
e=a1[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2', 'O_3',
 'PM10', 'SO_2', 'TCH', 'TOL', 'station']]
```

In [631]:
```
f=e.iloc[:,0:14]
g=e.iloc[:,-1]
```

In [632]:
```
h=StandardScaler().fit_transform(f)
```

In [633]:
```
logr=LogisticRegression(max_iter=10000)
logr.fit(h,g)
```

Out[633]:
```
LogisticRegression(max_iter=10000)
```

In [634]:

```
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 [635]:

```
i=[[10,20,30,40,50,60,15,26,37,47,58]]
```

In [636]:

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

```
[28079059]
```

In [637]:

```
logr.classes_
```

Out[637]:

```
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 [638]:

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

Out[638]:

```
0.0
```

In [639]:

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

Out[639]:

```
0.0
```

In [640]:

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

Out[640]:

```
0.9485714285714286
```

In [641]:

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

Out[641]:

```
ElasticNet()
```

In [642]:

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

```
[-0.          0.         -0.          0.97442165  0.        ]
```

In [643]:

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

```
1.3838448105340149
```

In [644]:

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

```
0.0
```

```
In [645]: from sklearn.ensemble import RandomForestClassifier
          rfc=RandomForestClassifier()
          rfc.fit(h_train,g_train)

Out[645]: RandomForestClassifier()
```

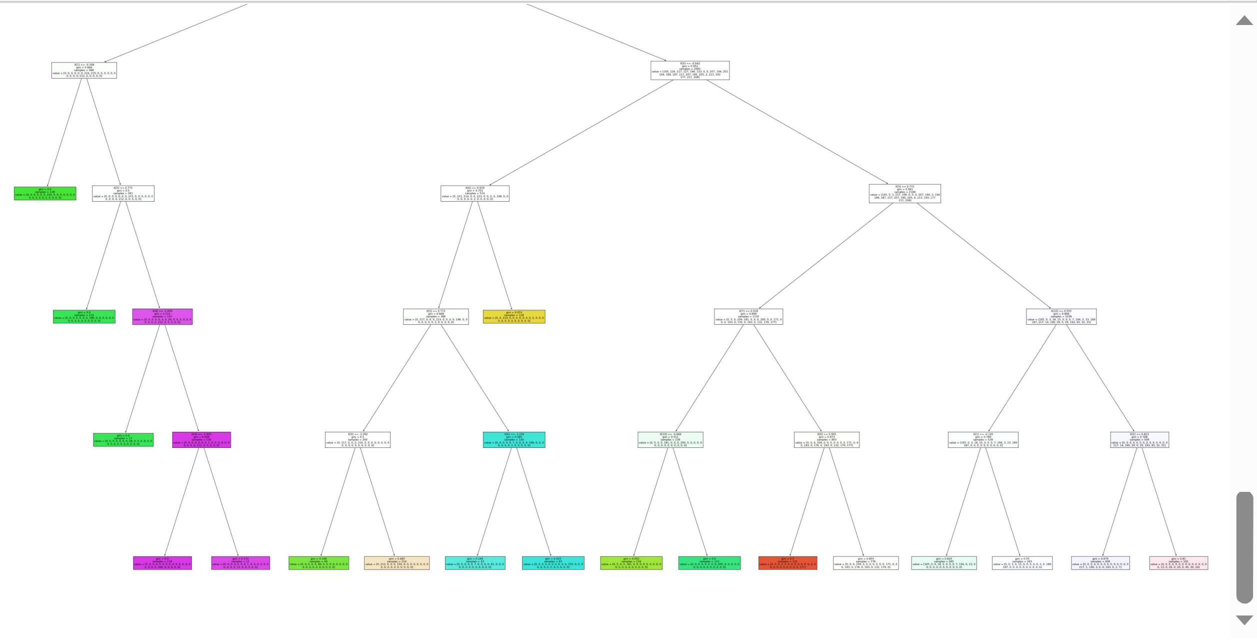
```
In [646]: parameters={'max_depth':[1,2,3,4,5],
           'min_samples_leaf':[5,10,15,20,25],
           'n_estimators':[10,20,30,40,50]
           }
```

```
In [647]: 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[647]: 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 [648]: grid_search.best_score_

Out[648]: 0.9977551020408163
```

```
In [649]: rfc_best=grid_search.best_estimator_
```

```
In [651]: from sklearn.tree import plot_tree
          plt.figure(figsize=(80,50))
          plot_tree(rfc_best.estimators_[2],filled=True)
```



Conclusion: from this data set i observed that the ridge has the highest accuracy of 0.9961224489795919

In [ ]: