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	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2001-08-01 01:00:00	NaN	0.37	NaN	NaN	NaN	58.400002	87.150002	NaN	34.529999	10
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	10
2	2001-08-01 01:00:00	NaN	0.28	NaN	NaN	NaN	50.660000	61.380001	NaN	46.310001	10
3	2001-08-01 01:00:00	NaN	0.47	NaN	NaN	NaN	69.790001	73.449997	NaN	40.650002	6
4	2001-08-01 01:00:00	NaN	0.39	NaN	NaN	NaN	22.830000	24.799999	NaN	66.309998	7
...	...	...	...	...	...	...	...	...	...	...	
217867	2001-04-01 00:00:00	10.45	1.81	NaN	NaN	NaN	73.000000	264.399994	NaN	5.200000	4
217868	2001-04-01 00:00:00	5.20	0.69	4.56	NaN	0.13	71.080002	129.300003	NaN	13.460000	2
217869	2001-04-01 00:00:00	0.49	1.09	NaN	1.00	0.19	76.279999	128.399994	0.35	5.020000	4
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.38	NaN	80.019997	197.000000	2.58	5.840000	3
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.04	0.18	76.809998	206.300003	5.20	8.340000	3

217872 rows × 16 columns

In [3]:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 217872 entries, 0 to 217871
Data columns (total 16 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     217872 non-null  object 
 1   BEN      70389 non-null   float64
 2   CO       216341 non-null  float64
 3   EBE      57752 non-null   float64
 4   MXY      42753 non-null   float64
 5   NMHC     85719 non-null   float64
 6   NO_2     216331 non-null  float64
 7   NOx      216318 non-null  float64
 8   OXY      42856 non-null   float64
 9   O_3      216514 non-null  float64
 10  PM10     207776 non-null  float64
 11  PXY      42845 non-null   float64
 12  SO_2     216403 non-null  float64
 13  TCH      85797 non-null   float64
 14  TOL      70196 non-null   float64
 15  station  217872 non-null  int64  
dtypes: float64(14), int64(1), object(1)
memory usage: 26.6+ MB
```

In [4]:
```
b=a.fillna(value=333)
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_
0	2001-08-01 01:00:00	333.00	0.37	333.00	333.00	333.00	58.400002	87.150002	333.00	34.52999
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.16000
2	2001-08-01 01:00:00	333.00	0.28	333.00	333.00	333.00	50.660000	61.380001	333.00	46.31000
3	2001-08-01 01:00:00	333.00	0.47	333.00	333.00	333.00	69.790001	73.449997	333.00	40.65000
4	2001-08-01 01:00:00	333.00	0.39	333.00	333.00	333.00	22.830000	24.799999	333.00	66.30999
...	...	...	...	...	...	...	...	...	...	
217867	2001-04-01 00:00:00	10.45	1.81	333.00	333.00	333.00	73.000000	264.399994	333.00	5.20000
217868	2001-04-01 00:00:00	5.20	0.69	4.56	333.00	0.13	71.080002	129.300003	333.00	13.46000
217869	2001-04-01 00:00:00	0.49	1.09	333.00	1.00	0.19	76.279999	128.399994	0.35	5.02000
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.38	333.00	80.019997	197.000000	2.58	5.84000
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.04	0.18	76.809998	206.300003	5.20	8.34000

217872 rows × 16 columns

In [5]:

Out[5]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3
',
       'PM10', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

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

Out[6]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2001-08-01 01:00:00	333.00	0.37	333.00	333.00	333.00	58.400002	87.150002	333.00	34.529999	1
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	1
2	2001-08-01 01:00:00	333.00	0.28	333.00	333.00	333.00	50.660000	61.380001	333.00	46.310001	1
3	2001-08-01 01:00:00	333.00	0.47	333.00	333.00	333.00	69.790001	73.449997	333.00	40.650002	
4	2001-08-01 01:00:00	333.00	0.39	333.00	333.00	333.00	22.830000	24.799999	333.00	66.309998	
5	2001-08-01 01:00:00	2.11	0.63	2.48	5.94	0.05	66.260002	118.099998	3.15	33.500000	1
6	2001-08-01 01:00:00	333.00	0.28	333.00	333.00	333.00	35.799999	39.590000	333.00	68.250000	1
7	2001-08-01 01:00:00	333.00	0.67	333.00	333.00	333.00	74.830002	112.000000	333.00	26.410000	1
8	2001-08-01 01:00:00	333.00	0.41	333.00	333.00	333.00	33.209999	37.299999	333.00	62.299999	1
9	2001-08-01 01:00:00	333.00	0.17	333.00	333.00	0.13	24.129999	36.970001	333.00	46.200001	
10	2001-08-01 01:00:00	333.00	0.38	333.00	333.00	0.02	40.900002	46.610001	333.00	51.450001	1

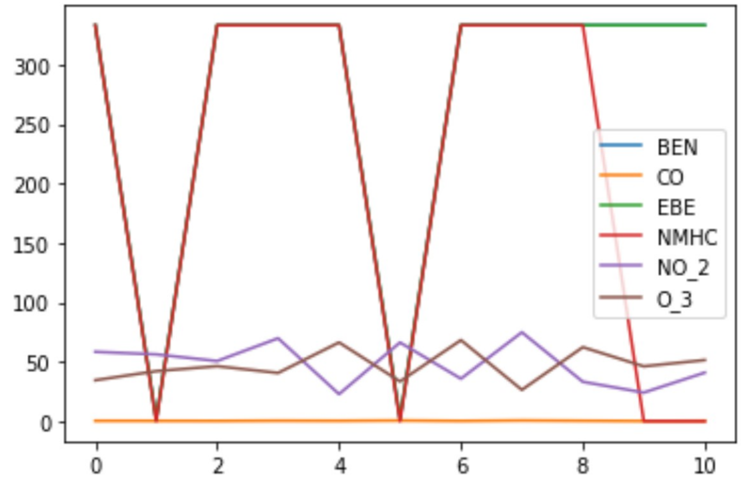
In [7]:
```
d=c[['BEN','CO','EBE','NMHC','NO_2','O_3']]
```

Out[7]:

	BEN	CO	EBE	NMHC	NO_2	O_3
0	333.00	0.37	333.00	333.00	58.400002	34.529999
1	1.50	0.34	1.49	0.07	56.250000	42.160000
2	333.00	0.28	333.00	333.00	50.660000	46.310001
3	333.00	0.47	333.00	333.00	69.790001	40.650002
4	333.00	0.39	333.00	333.00	22.830000	66.309998
5	2.11	0.63	2.48	0.05	66.260002	33.500000
6	333.00	0.28	333.00	333.00	35.799999	68.250000
7	333.00	0.67	333.00	333.00	74.830002	26.410000
8	333.00	0.41	333.00	333.00	33.209999	62.299999
9	333.00	0.17	333.00	0.13	24.129999	46.200001
10	333.00	0.38	333.00	0.02	40.900002	51.450001

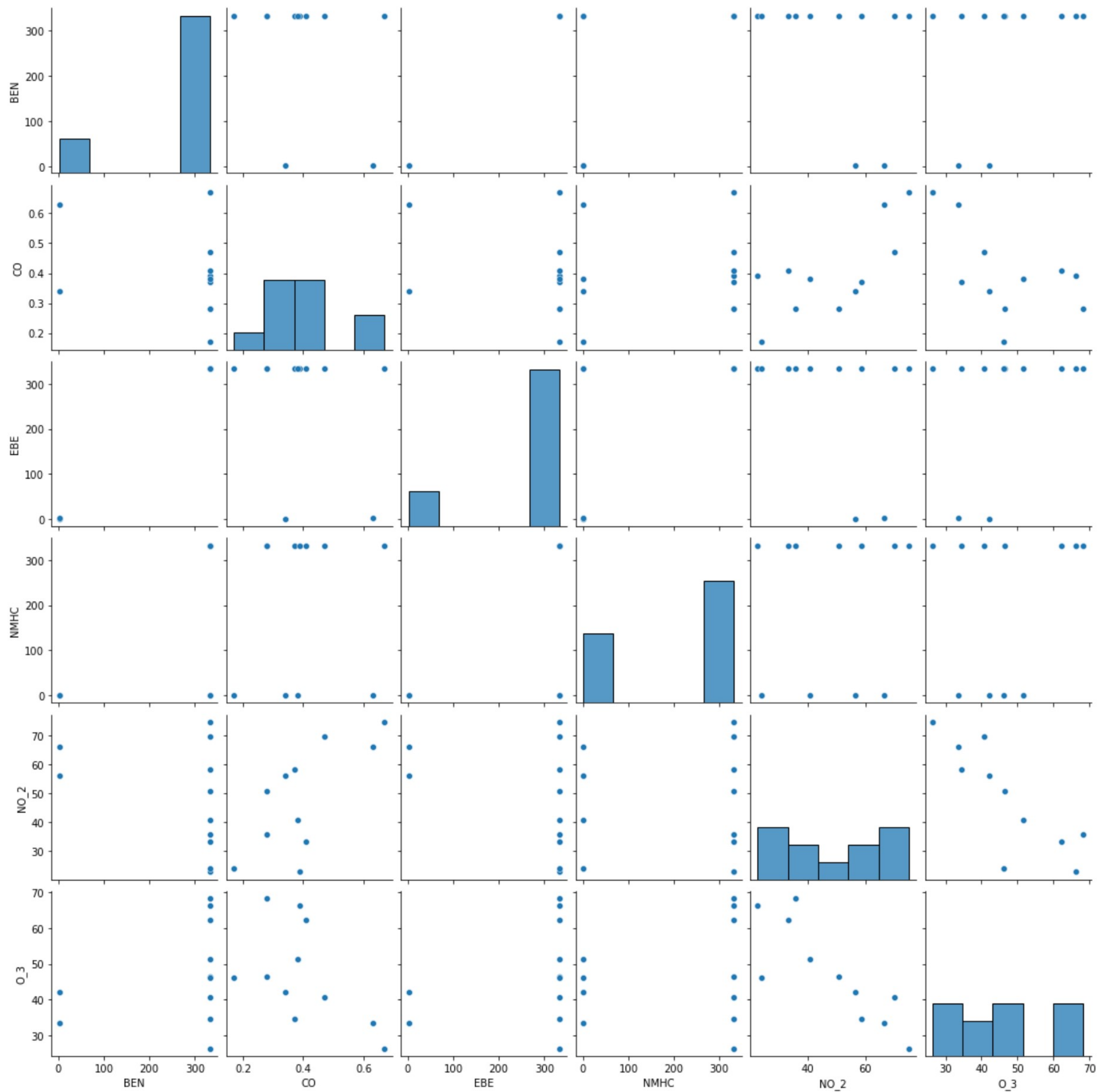
In [8]:
```
df.plot.line()
```

Out[8]:
```
<AxesSubplot:>
```



In [9]:

Out[9]: <seaborn.axisgrid.PairGrid at 0x287ba06b3a0>



In [10]:
```python
x=d[['BEN', 'CO', 'EBE', 'NMHC', 'NO_2']]
```

In [11]:
```python
from sklearn.model_selection import train_test_split
```

In [12]:
```python
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
```

Out[12]: LinearRegression()

In [13]:

-1.0341727474383333e-13

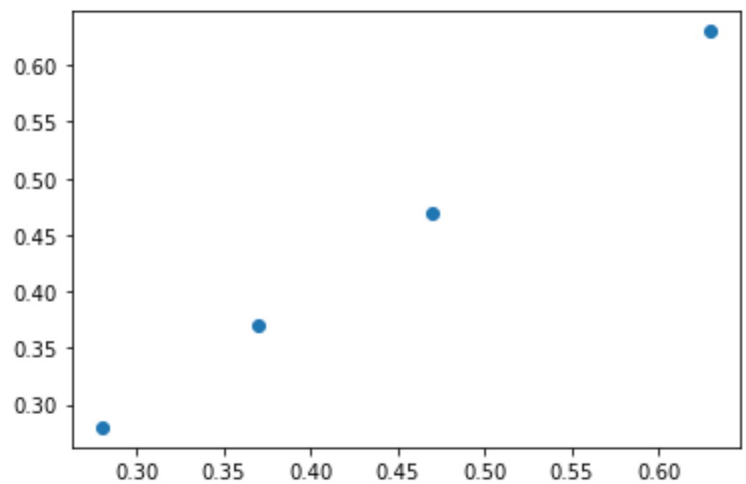
In [14]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
```

Out[14]:

	Co-efficient
BEN	2.563701e-14
CO	1.000000e+00
EBE	-2.540517e-14
NMHC	7.820330e-17
NO_2	-3.837914e-16

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

Out[15]:
```
<matplotlib.collections.PathCollection at 0x287bb1fd040>
```



In [16]:

```
1.0
```

In [17]:

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

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

In [19]:

Out[19]:
```
-1.271256674204904
```

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

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

In [21]:

Out[21]:
```
-0.3006861214158223
```

In [22]:
```python
a1=b.head(6500)
```

Out[22]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2001-08-01 01:00:00	333.00	0.37	333.00	333.0	333.00	58.400002	87.150002	333.00	34.529999	1
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.1	0.07	56.250000	75.169998	2.11	42.160000	1
2	2001-08-01 01:00:00	333.00	0.28	333.00	333.0	333.00	50.660000	61.380001	333.00	46.310001	1
3	2001-08-01 01:00:00	333.00	0.47	333.00	333.0	333.00	69.790001	73.449997	333.00	40.650002	
4	2001-08-01 01:00:00	333.00	0.39	333.00	333.0	333.00	22.830000	24.799999	333.00	66.309998	
...	...	...	...	...	...	...	...	...	...	...	
6495	2001-08-12 07:00:00	333.00	0.02	333.00	333.0	0.04	24.030001	25.850000	333.00	23.820000	
6496	2001-08-12 07:00:00	333.00	0.19	333.00	333.0	333.00	36.919998	41.060001	333.00	38.169998	
6497	2001-08-12 07:00:00	333.00	0.22	333.00	333.0	333.00	29.080000	35.060001	333.00	35.570000	
6498	2001-08-12 07:00:00	333.00	0.16	333.00	333.0	333.00	21.020000	23.549999	333.00	45.139999	
6499	2001-08-12 07:00:00	0.71	0.61	333.00	333.0	333.00	41.200001	51.410000	333.00	30.809999	

6500 rows × 16 columns

In [23]:
```python
e=a1[['BEN', 'CO', 'EBE','NMHC', 'NO_2','O_3',
```

In [24]:
```python
f=e.iloc[:,0:14]
```

In [25]:
```python
```

In [26]:
```python
logr=LogisticRegression(max_iter=10000)
```

Out[26]: LogisticRegression(max_iter=10000)

In [27]:
```python
from sklearn.model_selection import train_test_split
```

In [28]:
```python
```

In [29]:
```python
prediction=logr.predict(i)
```

```
[28079039]
```

In [30]:

Out[30]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079009,
       28079011, 28079012, 28079014, 28079015, 28079016, 28079018,
       28079019, 28079021, 28079022, 28079023, 28079024, 28079025,
       28079035, 28079036, 28079038, 28079039, 28079040, 28079099],
      dtype=int64)
```

In [31]:

Out[31]:
```
0.0
```

In [32]:

Out[32]:
```
0.0
```

In [33]:

Out[33]:
```
0.8682051282051282
```

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

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

In [35]:

```
[1.02462182e-04 0.00000000e+00 3.82222044e-08 3.53327940e-04
 4.61591830e-03]
```

In [36]:

```
0.09074088811121622
```

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

```
-0.8094046147861065
```

In [38]:
```python
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

Out[38]:
```
RandomForestClassifier()
```

In [39]:
```python
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]:

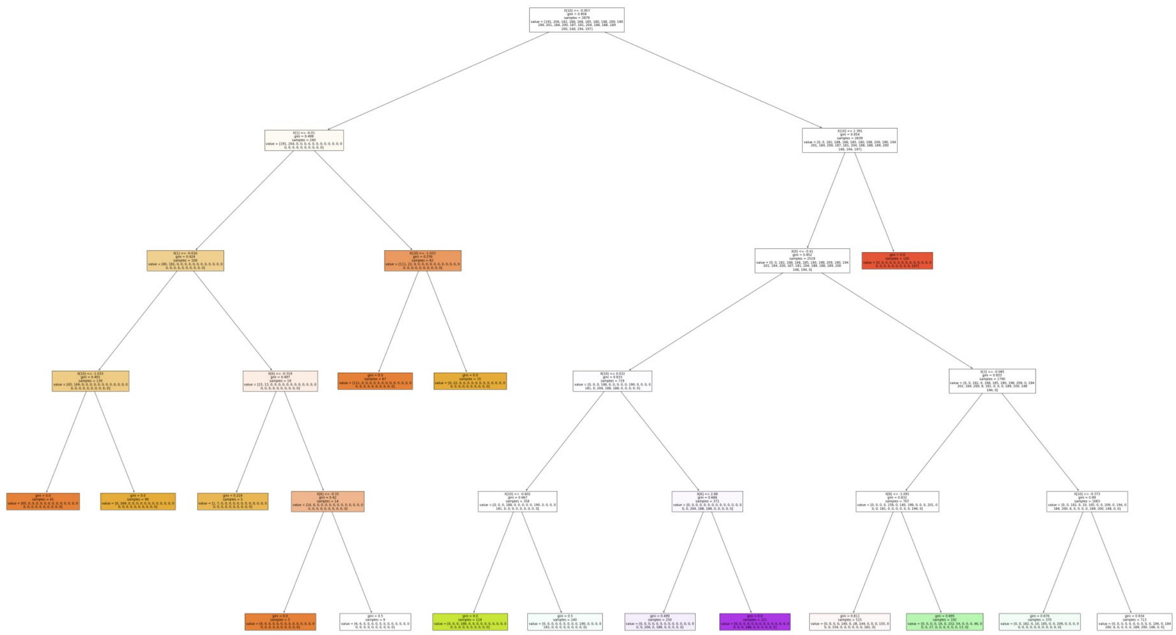
Out[41]:
```
0.9621978021978022
```

In [42]:

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

Out[43]:
```
[Text(2187.36, 2491.5, 'X[10] <= -0.957\ngini = 0.958\nsamples = 2879\nvalue
= [191, 204, 182, 188, 168, 185, 180, 198, 209, 190\n194, 201, 184, 200, 187,
181, 204, 188, 188, 189\n200, 148, 194, 197]'),
 Text(1160.64, 2038.5, 'X[1] <= -0.01\ngini = 0.499\nsamples = 240\nvalue =
[191, 204, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(714.24, 1585.5, 'X[1] <= -0.016\ngini = 0.424\nsamples = 158\nvalue =
[80, 182, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(357.12, 1132.5, 'X[10] <= -1.033\ngini = 0.401\nsamples = 139\nvalue =
[65, 169, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(178.56, 679.5, 'gini = 0.0\nsamples = 41\nvalue = [65, 0, 0, 0, 0, 0,
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 = 98\nvalue = [0, 169,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1071.3600000000001, 1132.5, 'X[6] <= -0.319\ngini = 0.497\nsamples = 1
9\nvalue = [15, 13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(892.8, 679.5, 'gini = 0.219\nsamples = 5\nvalue = [1, 7, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1249.92, 679.5, 'X[6] <= -0.15\ngini = 0.42\nsamples = 14\nvalue = [14,
6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1071.3600000000001, 226.5, 'gini = 0.0\nsamples = 5\nvalue = [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(1428.48, 226.5, 'gini = 0.5\nsamples = 9\nvalue = [6, 6, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1607.04, 1585.5, 'X[10] <= -1.033\ngini = 0.276\nsamples = 82\nvalue =
[111, 22, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1428.48, 1132.5, 'gini = 0.0\nsamples = 67\nvalue = [111, 0, 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, 'gini = 0.0\nsamples = 15\nvalue = [0, 22, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(3214.08, 2038.5, 'X[10] <= 2.391\ngini = 0.954\nsamples = 2639\nvalue =
[0, 0, 182, 188, 168, 185, 180, 198, 209, 190, 194\n201, 184, 200, 187, 181,
204, 188, 188, 189, 200\n148, 194, 197]'),
 Text(3035.52, 1585.5, 'X[0] <= -0.41\ngini = 0.952\nsamples = 2519\nvalue =
[0, 0, 182, 188, 168, 185, 180, 198, 209, 190, 194\n201, 184, 200, 187, 181,
204, 188, 188, 189, 200\n148, 194, 0]'),
 Text(2321.28, 1132.5, 'X[10] <= 0.032\ngini = 0.833\nsamples = 729\nvalue =
[0, 0, 0, 188, 0, 0, 0, 0, 0, 190, 0, 0, 0, 0\n181, 0, 204, 188, 188, 0, 0,
0, 0, 0]'),
 Text(1964.16, 679.5, 'X[10] <= -0.602\ngini = 0.667\nsamples = 358\nvalue =
[0, 0, 0, 188, 0, 0, 0, 0, 0, 190, 0, 0, 0, 0\n181, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1785.6, 226.5, 'gini = 0.0\nsamples = 118\nvalue = [0, 0, 0, 188, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2142.7200000000003, 226.5, 'gini = 0.5\nsamples = 240\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 190, 0, 0, 0, 0\n181, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2678.4, 679.5, 'X[6] <= 2.68\ngini = 0.666\nsamples = 371\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 204, 188, 188, 0, 0, 0, 0, 0]'),
 Text(2499.84, 226.5, 'gini = 0.499\nsamples = 250\nvalue = [0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 204, 0, 188, 0, 0, 0, 0, 0]'),
 Text(2856.96, 226.5, 'gini = 0.0\nsamples = 121\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 188, 0, 0, 0, 0, 0, 0]'),
 Text(3749.76, 1132.5, 'X[3] <= -0.085\ngini = 0.933\nsamples = 1790\nvalue =
[0, 0, 182, 0, 168, 185, 180, 198, 209, 0, 194\n201, 184, 200, 6, 181, 0, 0,
0, 189, 200, 148\n194, 0]'),
 Text(3392.64, 679.5, 'X[8] <= -1.091\ngini = 0.832\nsamples = 707\nvalue =
[0, 0, 0, 0, 158, 0, 180, 198, 0, 0, 0, 201, 0\n0, 0, 181, 0, 0, 0, 0, 0, 0,
194, 0]'),
 Text(3214.08, 226.5, 'gini = 0.811\nsamples = 515\nvalue = [0, 0, 0, 0, 140,
0, 28, 144, 0, 0, 0, 155, 0\n0, 0, 154, 0, 0, 0, 0, 0, 0, 181, 0]'),
 Text(3571.2, 226.5, 'gini = 0.695\nsamples = 192\nvalue = [0, 0, 0, 0, 18,
0, 152, 54, 0, 0, 0, 46, 0\n0, 0, 27, 0, 0, 0, 0, 0, 0, 13, 0]'),
 Text(4106.88, 679.5, 'X[10] <= -0.373\ngini = 0.89\nsamples = 1083\nvalue =
[0, 0, 182, 0, 10, 185, 0, 0, 209, 0, 194, 0\n184, 200, 6, 0, 0, 0, 0, 189, 2
00, 148, 0, 0]'),
 Text(3928.32, 226.5, 'gini = 0.676\nsamples = 370\nvalue = [0, 0, 182, 0, 1
0, 185, 0, 0, 209, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(4285.4400000000005, 226.5, 'gini = 0.834\nsamples = 713\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 194, 0, 184\n200, 6, 0, 0, 0, 0, 189, 200, 148, 0,
0]'),
 Text(3392.64, 1585.5, 'gini = 0.0\nsamples = 120\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 197]')]
```



From this observation I had observe that the GRIDSEARCH is a highest accuracy of 0.9621978021978022

In [ ]: