In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as py
import seaborn as sns
from sklearn.linear_model import LogisticRegression
```

In [2]:

```
df =pd.read_csv(r"D:\datasets\madrid_2008.csv")
df
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY
0	2008-06-01 01:00:00	NaN	0.47	NaN	NaN	NaN	83.089996	120.699997	NaN
1	2008-06-01 01:00:00	NaN	0.59	NaN	NaN	NaN	94.820000	130.399994	NaN
2	2008-06-01 01:00:00	NaN	0.55	NaN	NaN	NaN	75.919998	104.599998	NaN
3	2008-06-01 01:00:00	NaN	0.36	NaN	NaN	NaN	61.029999	66.559998	NaN
4	2008-06-01 01:00:00	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61
...	...	...	...	...	...	...	...	...	...
226387	2008-11-01 00:00:00	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91
226388	2008-11-01 00:00:00	NaN	0.30	NaN	NaN	NaN	41.880001	48.500000	NaN
226389	2008-11-01 00:00:00	0.25	NaN	0.56	NaN	0.11	83.610001	102.199997	NaN
226390	2008-11-01 00:00:00	0.54	NaN	2.70	NaN	0.18	70.639999	81.860001	NaN
226391	2008-11-01 00:00:00	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64

226392 rows × 17 columns

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 226392 entries, 0 to 226391
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     226392 non-null  object 
 1   BEN      67047 non-null   float64
 2   CO       208109 non-null  float64
 3   EBE      67044 non-null   float64
 4   MXY      25867 non-null   float64
 5   NMHC     85079 non-null   float64
 6   NO_2     225315 non-null  float64
 7   NOx      225311 non-null  float64
 8   OXY      25878 non-null   float64
 9   O_3      215716 non-null  float64
 10  PM10     220179 non-null  float64
 11  PM25     67833 non-null   float64
 12  PXY      25877 non-null   float64
 13  SO_2     225405 non-null  float64
 14  TCH      85107 non-null   float64
 15  TOL      66940 non-null   float64
 16  station  226392 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 29.4+ MB
```

In [4]:
```
df1 =df.fillna(value=0)
df1
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY
0	2008-06-01 01:00:00	0.00	0.47	0.00	0.00	0.00	83.089996	120.699997	0.00
1	2008-06-01 01:00:00	0.00	0.59	0.00	0.00	0.00	94.820000	130.399994	0.00
2	2008-06-01 01:00:00	0.00	0.55	0.00	0.00	0.00	75.919998	104.599998	0.00
3	2008-06-01 01:00:00	0.00	0.36	0.00	0.00	0.00	61.029999	66.559998	0.00
4	2008-06-01 01:00:00	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61
...	...	...	...	...	...	...	...	...	...
226387	2008-11-01 00:00:00	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91
226388	2008-11-01 00:00:00	0.00	0.30	0.00	0.00	0.00	41.880001	48.500000	0.00
226389	2008-11-01 00:00:00	0.25	0.00	0.56	0.00	0.11	83.610001	102.199997	0.00
226390	2008-11-01 00:00:00	0.54	0.00	2.70	0.00	0.18	70.639999	81.860001	0.00
226391	2008-11-01 00:00:00	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64

226392 rows × 17 columns

In [5]:

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 226392 entries, 0 to 226391
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     226392 non-null  object 
 1   BEN      226392 non-null  float64
 2   CO       226392 non-null  float64
 3   EBE      226392 non-null  float64
 4   MXY      226392 non-null  float64
 5   NMHC     226392 non-null  float64
 6   NO_2     226392 non-null  float64
 7   NOx      226392 non-null  float64
 8   OXY      226392 non-null  float64
 9   O_3      226392 non-null  float64
 10  PM10     226392 non-null  float64
 11  PM25     226392 non-null  float64
 12  PXY      226392 non-null  float64
 13  SO_2     226392 non-null  float64
 14  TCH      226392 non-null  float64
 15  TOL      226392 non-null  float64
 16  station  226392 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 29.4+ MB
```

In [6]:
```
df1.columns
```

Out[6]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_
3',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [7]:
```
df2=df1[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station']]
df2
```

Out[7]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
0	0.00	0.47	0.00	0.00	0.00	83.089996	120.699997	0.00	16.990000
1	0.00	0.59	0.00	0.00	0.00	94.820000	130.399994	0.00	17.469999
2	0.00	0.55	0.00	0.00	0.00	75.919998	104.599998	0.00	13.470000
3	0.00	0.36	0.00	0.00	0.00	61.029999	66.559998	0.00	23.110001
4	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61	12.120000
...	...	...	...	...	...	...	...	...	...
226387	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91	57.400002
226388	0.00	0.30	0.00	0.00	0.00	41.880001	48.500000	0.00	35.830002
226389	0.25	0.00	0.56	0.00	0.11	83.610001	102.199997	0.00	14.130000
226390	0.54	0.00	2.70	0.00	0.18	70.639999	81.860001	0.00	0.000000
226391	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64	31.910000

226392 rows × 16 columns

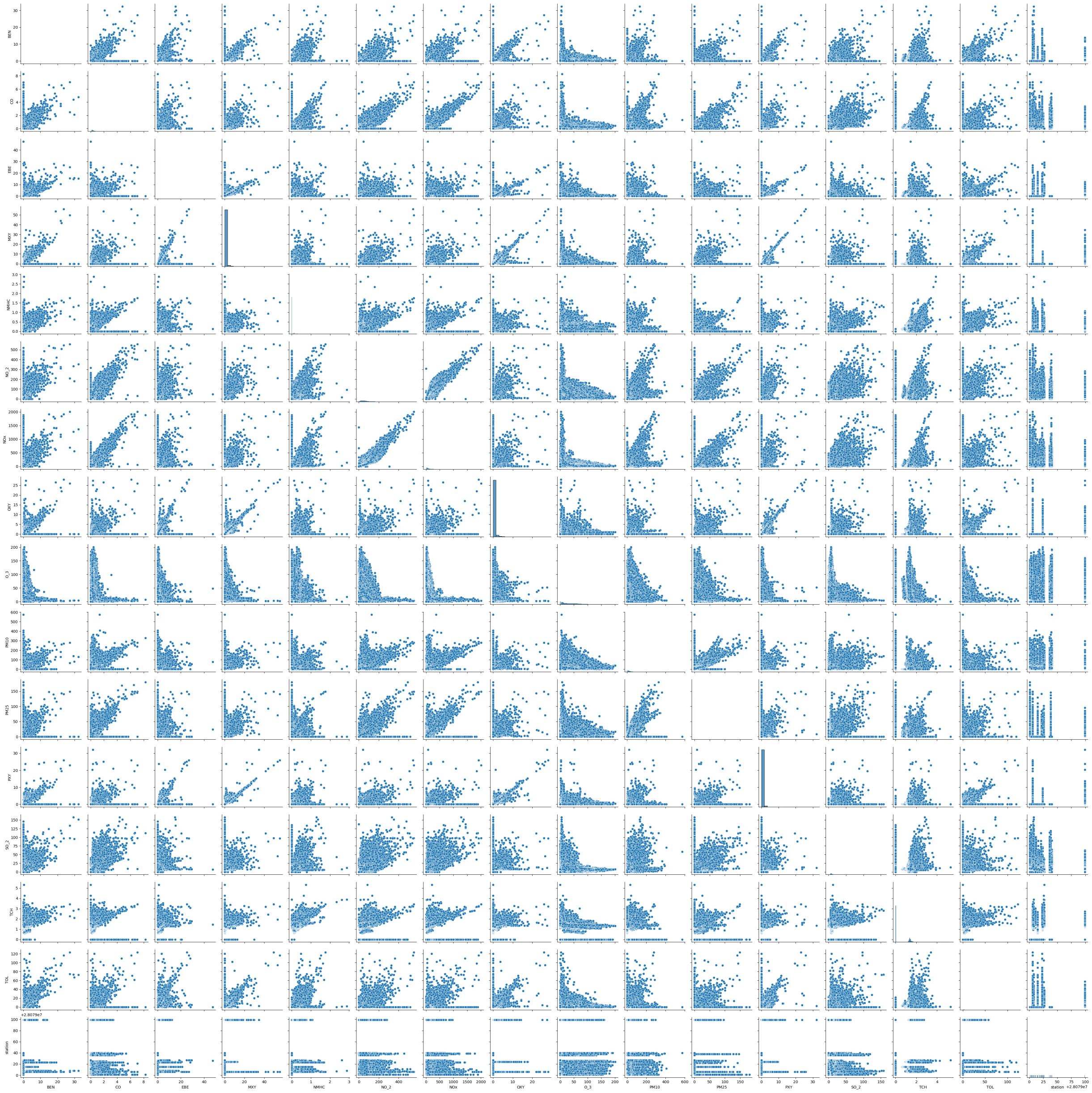
In [8]:

```
sns.pairplot(df2)
```

```
C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\seabor
n\axisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

Out[8]:

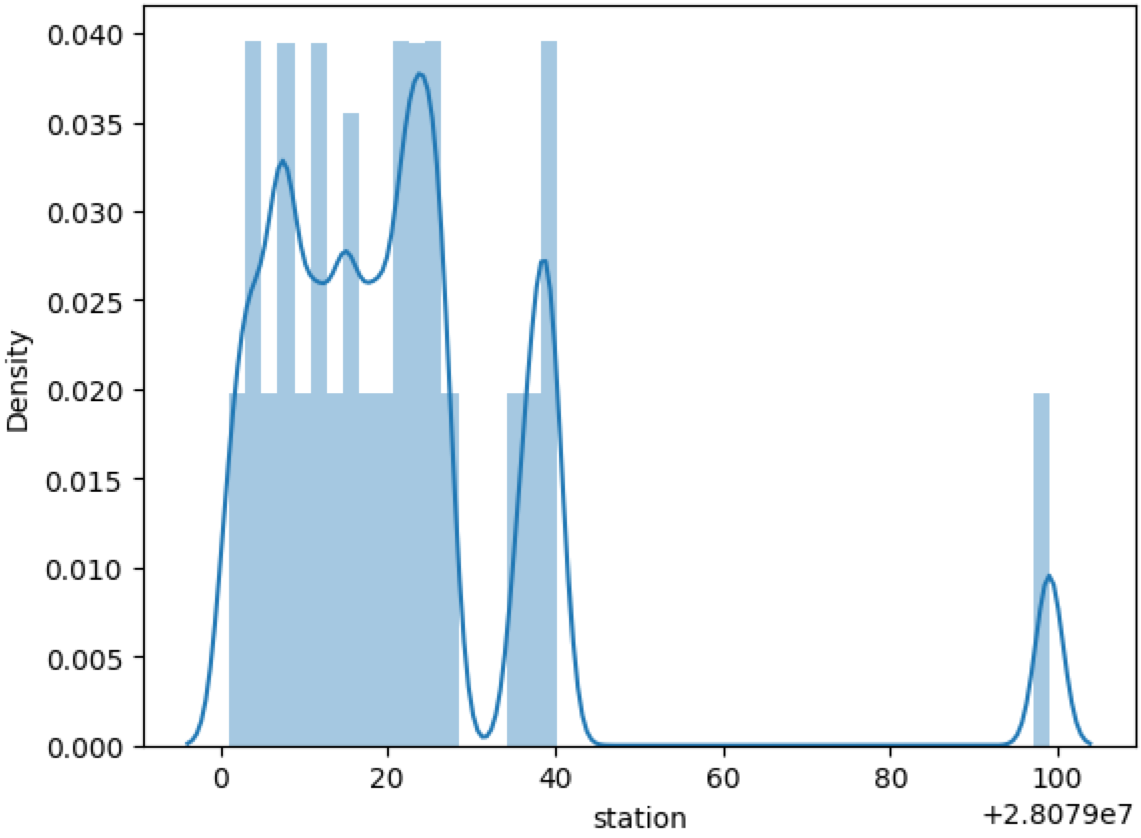
```
<seaborn.axisgrid.PairGrid at 0x20cbf1b1b10>
```



```
In [9]: sns.distplot(df2['station'])
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_18380\1070072814.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(df2['station'])
```

```
Out[9]: <Axes: xlabel='station', ylabel='Density'>
```



```
In [10]: x=df2[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
                'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL']]
         y=df2['station']
```

```
In [11]: 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)
```

linear

```
In [12]: from sklearn.linear_model import LinearRegression
```

```
In [13]: lr=LinearRegression()
         lr.fit(x_train,y_train)
```

```
Out[13]: ▼ LinearRegression
         LinearRegression()
```

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

Out[14]:

	Co-efficient
BEN	-3.354127
CO	-10.509262
EBE	1.727078
MXY	-1.251906
NMHC	0.752771
NO_2	-0.070672
NOx	0.024686
OXY	3.808906
O_3	-0.030009
PM10	0.012944
PM25	0.241430
PXY	9.280505
SO_2	-0.090592
TCH	0.835774
TOL	-0.398473

In [15]:

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

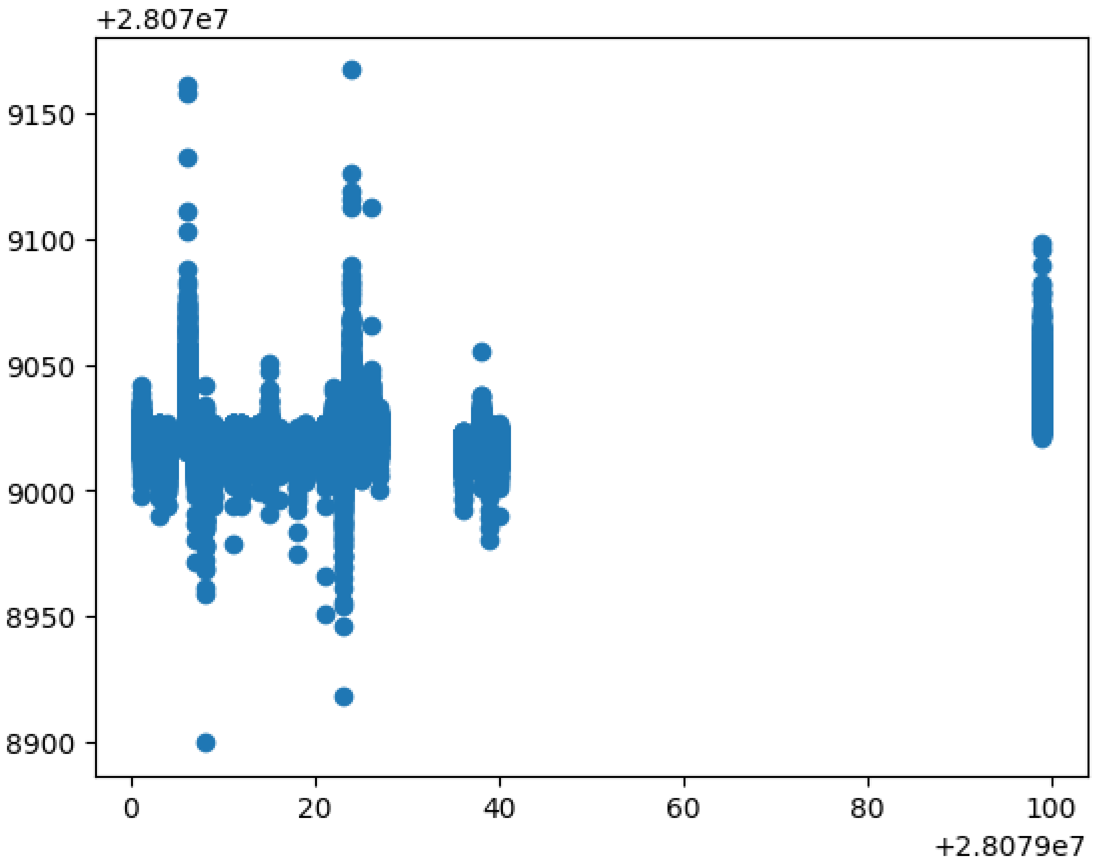
```
28079026.48541442
```

In [16]:

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

Out[16]:

```
<matplotlib.collections.PathCollection at 0x20c87028290>
```



In [17]:

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

```
0.1402608789458767
```

In [18]:

```
print(lr.score(x_train,y_train))
```

```
0.13424011207710673
```

Ridge

In [19]:

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

In [20]:

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

Out[20]:

```
▼ Ridge
Ridge(alpha=10)
```

In [21]:

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

Out[21]:

```
0.1402434037616892
```

Lasso

```
In [22]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)
```

```
Out[22]: ▼      Lasso
         Lasso(alpha=10)
```

```
In [23]: la.score(x_test,y_test)
```

```
Out[23]: 0.045338066933052756
```

elasticnet

```
In [24]: from sklearn.linear_model import ElasticNet
         en=ElasticNet()
         en.fit(x_train,y_train)
```

```
Out[24]: ▼ ElasticNet
         ElasticNet()
```

```
In [25]: print(en.coef_)
```

```
[-0.         -0.          0.          1.73457161  0.         -0.05936775
 -0.01442502  1.23684634 -0.03562738  0.01699235  0.3231309   1.01080816
 -0.15080874  0.31505937 -0.1074231 ]
```

```
In [26]: print(en.intercept_)
```

```
28079026.545498498
```

```
In [27]: print(en.predict(x_test))
```

```
[28079020.75269785 28079021.23305743 28079024.5408356  ...
 28079033.19802859 28079021.48640617 28079021.73647853]
```

```
In [28]: print(en.score(x_test,y_test))
```

```
0.10169399736788343
```

logistic

```
In [29]: feature_matrix =df2.iloc[:,0:15]
         target_vector=df2.iloc[:,-1]
```

In [30]:
```
feature_matrix=df2[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL']]
y=df2['station']
```

In [31]:
```
feature_matrix.shape
```

Out[31]:
```
(226392, 15)
```

In [32]:
```
target_vector.shape
```

Out[32]:
```
(226392,)
```

In [33]:
```
from sklearn.preprocessing import StandardScaler
```

In [34]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [35]:
```
logr = LogisticRegression()
logr.fit(fs,target_vector)
```

```
C:\Users\HP\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklear
n\linear_model\_logistic.py:460: ConvergenceWarning: lbfgs failed to converg
e (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion
  n_iter_i = _check_optimize_result(
```

Out[35]:
```
▼ LogisticRegression

LogisticRegression()
```

In [36]:
```
observation=[[1,2,3,4,5,6,7,8,9,11,12,13,14,15,16]]
```

In [37]:
```
prediction =logr.predict(observation)
print(prediction)
```

```
[28079099]
```

In [38]:
```
logr.classes_
```

Out[38]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079018, 28079019, 28079021, 28079022, 28079023, 28079024,
       28079025, 28079026, 28079027, 28079036, 28079038, 28079039,
       28079040, 28079099], dtype=int64)
```

In [39]:
```
logr.score(fs,target_vector)
```

Out[39]:
```
0.49849376303049575
```

In [40]:
```
logr.predict_proba(observation)[0][0]
```

Out[40]:
```
6.191323914672894e-109
```

In [41]:
```
logr.predict_proba(observation)[0][1]
```

Out[41]:
```
1.5163026503883967e-162
```

Random forest

In [42]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import plot_tree
```

In [43]:
```
x=df2.drop('station',axis=1)
y=df2['station']
```

In [44]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.70)
```

In [45]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[45]:
```
▼ RandomForestClassifier
RandomForestClassifier()
```

In [46]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf' :[6,7,8,9,10],
            'n_estimators':[11,12,13,14,15]}
```

In [47]:
```
from sklearn.model_selection import GridSearchCV
```

In [55]:
```
grid_search =GridSearchCV(estimator =rfc,param_grid=parameters,cv=2,scoring=
grid_search.fit(x_train,y_train)
```

Out[55]:
```
▶              GridSearchCV
▶ estimator: RandomForestClassifier
      ▶ RandomForestClassifier
```

In [56]:
```
grid_search.best_score_
```

Out[56]:
```
0.49461848194031977
```

In [57]:
```
rfc_best=grid_search.best_estimator_
```

In [51]:
```
py.figure(figsize=(80,50))
plot_tree(rfc_best.estimators_[5],filled=True)
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

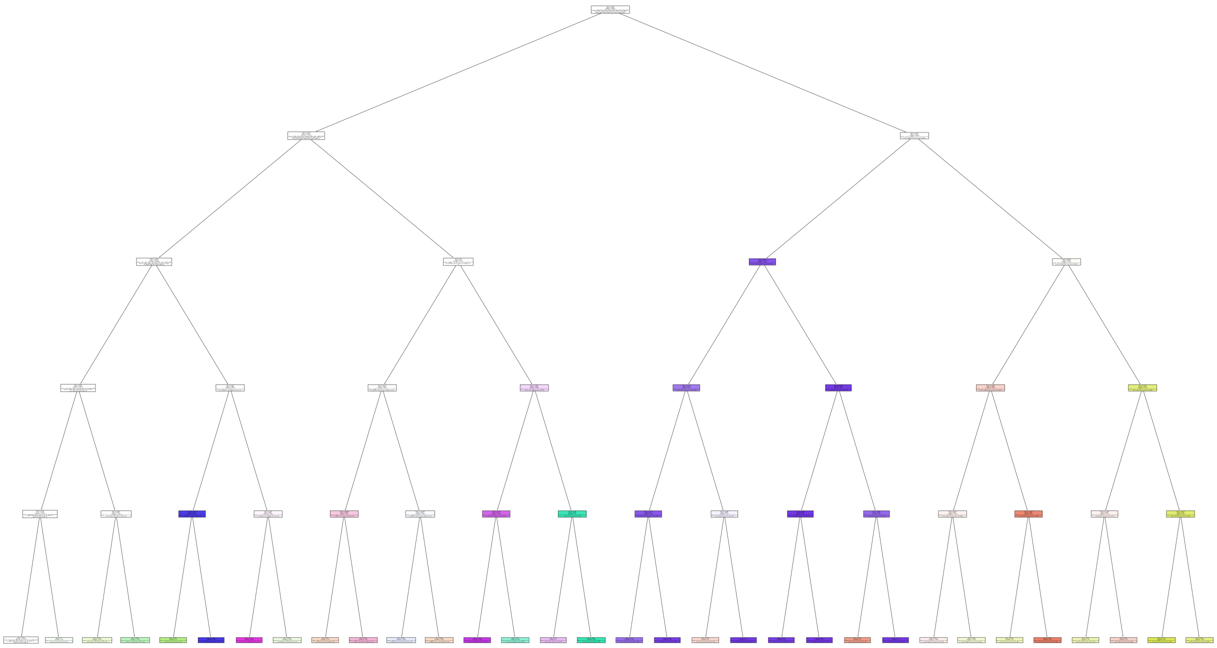
```
Out[51]: [Text(0.5, 0.9166666666666666, 'x[11] <= 0.08\ngini = 0.961\nsamples = 4281
         6\nvalue = [2644, 2671, 2672, 2699, 2626, 2591, 2707, 2609, 2685\n2523, 208
         3, 2608, 2635, 2619, 2552, 2685, 2700, 2694\n2597, 2590, 2640, 2658, 2642,
         2554, 2580, 2653]'),
          Text(0.25, 0.75, 'x[10] <= 0.285\ngini = 0.957\nsamples = 37833\nvalue =
         [2644, 2671, 2672, 36, 2626, 2591, 2707, 2609, 2685\n2523, 2083, 2608, 263
         5, 2619, 2552, 2685, 2700, 79\n2597, 2590, 2640, 2658, 2642, 2554, 2580,
         1]'),
          Text(0.125, 0.5833333333333334, 'x[14] <= 0.055\ngini = 0.945\nsamples = 2
         9886\nvalue = [64, 2671, 2672, 16, 2626, 2591, 2707, 2609, 2685\n2523, 6, 2
         608, 2635, 2619, 2552, 6, 2700, 47, 2597\n13, 2640, 2658, 20, 2554, 2580,
         0]'),
          Text(0.0625, 0.4166666666666667, 'x[4] <= 0.005\ngini = 0.935\nsamples = 2
         5055\nvalue = [64, 2671, 2672, 16, 2626, 164, 2707, 2609, 2685\n2523, 3, 26
         08, 2635, 2619, 2552, 6, 52, 47, 2597\n7, 22, 2658, 20, 2554, 2580, 0]'),
          Text(0.03125, 0.25, 'x[13] <= 0.59\ngini = 0.924\nsamples = 21665\nvalue =
         [64, 2671, 2672, 16, 40, 31, 2707, 45, 2685, 2523\n3, 2608, 2635, 2619, 255
         2, 6, 13, 47, 2597, 7, 7\n2658, 20, 2554, 2580, 0]'),
          Text(0.015625, 0.08333333333333333, 'gini = 0.924\nsamples = 21639\nvalue
         = [64, 2671, 2672, 16, 20, 31, 2707, 24, 2685, 2523\n3, 2608, 2635, 2619, 2
         552, 6, 13, 47, 2597, 7, 7\n2658, 20, 2554, 2580, 0]'),
          Text(0.046875, 0.08333333333333333, 'gini = 0.5\nsamples = 26\nvalue = [0,
         0, 0, 0, 20, 0, 0, 21, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0]'),
          Text(0.09375, 0.25, 'x[8] <= 49.47\ngini = 0.534\nsamples = 3390\nvalue =
         [0, 0, 0, 0, 2586, 133, 0, 2564, 0, 0, 0, 0, 0\n0, 0, 0, 39, 0, 0, 0, 15,
         0, 0, 0, 0, 0]'),
          Text(0.078125, 0.08333333333333333, 'gini = 0.529\nsamples = 2176\nvalue =
         [0, 0, 0, 0, 1844, 82, 0, 1433, 0, 0, 0, 0, 0\n0, 0, 0, 32, 0, 0, 0, 15, 0,
         0, 0, 0, 0]'),
          Text(0.109375, 0.08333333333333333, 'gini = 0.509\nsamples = 1214\nvalue =
         [0, 0, 0, 0, 742, 51, 0, 1131, 0, 0, 0, 0, 0\n0, 0, 0, 7, 0, 0, 0, 0, 0, 0,
         0, 0, 0]'),
          Text(0.1875, 0.4166666666666667, 'x[9] <= 0.69\ngini = 0.667\nsamples = 48
         31\nvalue = [0, 0, 0, 0, 0, 2427, 0, 0, 0, 0, 3, 0, 0, 0\n0, 0, 2648, 0, 0,
         6, 2618, 0, 0, 0, 0, 0]'),
          Text(0.15625, 0.25, 'x[8] <= 7.675\ngini = 0.035\nsamples = 396\nvalue =
         [0, 0, 0, 0, 0, 11, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 598, 0, 0, 0, 0, 0, 0, 0,
         0, 0]'),
          Text(0.140625, 0.08333333333333333, 'gini = 0.408\nsamples = 8\nvalue =
         [0, 0, 0, 0, 0, 10, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 4, 0, 0, 0, 0, 0, 0, 0,
         0, 0]'),
          Text(0.171875, 0.08333333333333333, 'gini = 0.003\nsamples = 388\nvalue =
         [0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 594, 0, 0, 0, 0, 0, 0, 0,
         0, 0]'),
          Text(0.21875, 0.25, 'x[1] <= 0.06\ngini = 0.664\nsamples = 4435\nvalue =
         [0, 0, 0, 0, 0, 2416, 0, 0, 0, 0, 3, 0, 0, 0\n0, 0, 2050, 0, 0, 6, 2618, 0,
         0, 0, 0, 0]'),
          Text(0.203125, 0.08333333333333333, 'gini = 0.006\nsamples = 1650\nvalue =
         [0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 6, 2618, 0, 0, 0,
         0, 0]'),
          Text(0.234375, 0.08333333333333333, 'gini = 0.497\nsamples = 2785\nvalue =
         [0, 0, 0, 0, 0, 2414, 0, 0, 0, 0, 3, 0, 0, 0\n0, 0, 2050, 0, 0, 0, 0, 0, 0,
         0, 0, 0]'),
          Text(0.375, 0.5833333333333334, 'x[2] <= 0.05\ngini = 0.8\nsamples = 7947
 Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js, 0, 0, 0, 0, 0, 0, 2077, 0, 0\n0, 0, 2679, 0, 32,
0, 2577, 0, 0, 2622, 0, 0\n1]'),
 Text(0.3125, 0.4166666666666667, 'x[9] <= 10.14\ngini = 0.673\nsamples = 5
019\nvalue = [2580, 0, 0, 20, 0, 0, 0, 0, 0, 0, 13, 0, 0\n0, 0, 2679, 0, 3
2, 0, 9, 0, 0, 2622, 0, 0, 0]'),
 Text(0.28125, 0.25, 'x[8] <= 29.965\ngini = 0.646\nsamples = 671\nvalue =
[297, 0, 0, 6, 0, 0, 0, 0, 0, 0, 11, 0, 0, 0\n0, 256, 0, 1, 0, 1, 0, 0, 51
3, 0, 0, 0]'),
 Text(0.265625, 0.08333333333333333, 'gini = 0.64\nsamples = 127\nvalue =
[98, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 45, 0, 0, 0, 1, 0, 0, 58, 0,
0, 0]'),
 Text(0.296875, 0.08333333333333333, 'gini = 0.625\nsamples = 544\nvalue =
[199, 0, 0, 4, 0, 0, 0, 0, 0, 0, 11, 0, 0, 0\n0, 211, 0, 1, 0, 0, 0, 0, 45
5, 0, 0, 0]'),
 Text(0.34375, 0.25, 'x[10] <= 19.325\ngini = 0.671\nsamples = 4348\nvalue
= [2283, 0, 0, 14, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0\n0, 2423, 0, 31, 0, 8, 0,
0, 2109, 0, 0, 0]'),
 Text(0.328125, 0.08333333333333333, 'gini = 0.665\nsamples = 2903\nvalue =
[1203, 0, 0, 13, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 1815, 0, 31, 0, 7, 0, 0,
1477, 0, 0, 0]'),
 Text(0.359375, 0.08333333333333333, 'gini = 0.642\nsamples = 1445\nvalue =
[1080, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0\n0, 608, 0, 0, 0, 1, 0, 0, 63
2, 0, 0, 0]'),
 Text(0.4375, 0.4166666666666667, 'x[2] <= 1.005\ngini = 0.494\nsamples = 2
928\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2064, 0, 0, 0\n0, 0, 0, 0, 0, 2
568, 0, 0, 0, 0, 0, 1]'),
 Text(0.40625, 0.25, 'x[14] <= 2.06\ngini = 0.307\nsamples = 2014\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 597, 0, 0, 0\n0, 0, 0, 0, 0, 2559, 0, 0, 0,
0, 0, 0]'),
 Text(0.390625, 0.08333333333333333, 'gini = 0.013\nsamples = 1478\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15, 0, 0, 0\n0, 0, 0, 0, 0, 2299, 0, 0, 0,
0, 0, 0]'),
 Text(0.421875, 0.08333333333333333, 'gini = 0.427\nsamples = 536\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 582, 0, 0, 0\n0, 0, 0, 0, 0, 260, 0, 0, 0,
0, 0, 0]'),
 Text(0.46875, 0.25, 'x[0] <= 0.46\ngini = 0.013\nsamples = 914\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1467, 0, 0, 0\n0, 0, 0, 0, 0, 9, 0, 0, 0, 0,
0, 1]'),
 Text(0.453125, 0.08333333333333333, 'gini = 0.48\nsamples = 9\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0\n0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0,
0]'),
 Text(0.484375, 0.08333333333333333, 'gini = 0.001\nsamples = 905\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1461, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1]'),
 Text(0.75, 0.75, 'x[5] <= 22.07\ngini = 0.667\nsamples = 4983\nvalue = [0,
0, 0, 2663, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 2615, 0, 0, 0, 0, 0, 0,
0, 2652]'),
 Text(0.625, 0.5833333333333334, 'x[11] <= 0.835\ngini = 0.23\nsamples = 10
63\nvalue = [0, 0, 0, 75, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 1474, 0,
0, 0, 0, 0, 0, 0, 141]'),
 Text(0.5625, 0.4166666666666667, 'x[6] <= 23.315\ngini = 0.43\nsamples = 4
58\nvalue = [0, 0, 0, 73, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 541, 0, 0,
0, 0, 0, 0, 0, 129]'),
 Text(0.53125, 0.25, 'x[9] <= 11.035\ngini = 0.25\nsamples = 331\nvalue =
[0, 0, 0, 20, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 457, 0, 0, 0, 0, 0, 0,
0, 55]'),
 Text(0.515625, 0.08333333333333333, 'gini = 0.376\nsamples = 185\nvalue =
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
[0, 0, 0, 19, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 227, 0, 0, 0, 0, 0, 0,
0, 49]'),
 Text(0.546875, 0.08333333333333333, 'gini = 0.058\nsamples = 146\nvalue =
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 230, 0, 0, 0, 0, 0, 0,
0, 6]'),
 Text(0.59375, 0.25, 'x[4] <= 0.185\ngini = 0.655\nsamples = 127\nvalue =
[0, 0, 0, 53, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 84, 0, 0, 0, 0, 0, 0,
0, 74]'),
 Text(0.578125, 0.08333333333333333, 'gini = 0.602\nsamples = 85\nvalue =
[0, 0, 0, 53, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 21, 0, 0, 0, 0, 0, 0,
0, 74]'),
 Text(0.609375, 0.08333333333333333, 'gini = 0.0\nsamples = 42\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 63, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(0.6875, 0.4166666666666667, 'x[6] <= 23.955\ngini = 0.029\nsamples =
605\nvalue = [0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 933, 0, 0,
0, 0, 0, 0, 0, 12]'),
 Text(0.65625, 0.25, 'x[7] <= 0.955\ngini = 0.005\nsamples = 565\nvalue =
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 884, 0, 0, 0, 0, 0, 0,
0, 1]'),
 Text(0.640625, 0.08333333333333333, 'gini = 0.049\nsamples = 24\nvalue =
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 39, 0, 0, 0, 0, 0, 0,
0, 1]'),
 Text(0.671875, 0.08333333333333333, 'gini = 0.002\nsamples = 541\nvalue =
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 845, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(0.71875, 0.25, 'x[1] <= 0.265\ngini = 0.322\nsamples = 40\nvalue =
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 49, 0, 0, 0, 0, 0, 0,
0, 11]'),
 Text(0.703125, 0.08333333333333333, 'gini = 0.461\nsamples = 11\nvalue =
[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0,
11]'),
 Text(0.734375, 0.08333333333333333, 'gini = 0.0\nsamples = 29\nvalue = [0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 45, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(0.875, 0.5833333333333334, 'x[14] <= 6.805\ngini = 0.633\nsamples = 3
920\nvalue = [0, 0, 0, 2588, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 1141,
0, 0, 0, 0, 0, 0, 0, 2511]'),
 Text(0.8125, 0.4166666666666667, 'x[11] <= 1.265\ngini = 0.635\nsamples =
2804\nvalue = [0, 0, 0, 1301, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 1048,
0, 0, 0, 0, 0, 0, 0, 2123]'),
 Text(0.78125, 0.25, 'x[6] <= 65.35\ngini = 0.66\nsamples = 2143\nvalue =
[0, 0, 0, 1156, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 932, 0, 0, 0, 0, 0,
0, 0, 1338]'),
 Text(0.765625, 0.08333333333333333, 'gini = 0.632\nsamples = 1056\nvalue =
[0, 0, 0, 309, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 634, 0, 0, 0, 0, 0,
0, 0, 725]'),
 Text(0.796875, 0.08333333333333333, 'gini = 0.618\nsamples = 1087\nvalue =
[0, 0, 0, 847, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 298, 0, 0, 0, 0, 0,
0, 0, 613]'),
 Text(0.84375, 0.25, 'x[4] <= 0.105\ngini = 0.405\nsamples = 661\nvalue =
[0, 0, 0, 145, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 116, 0, 0, 0, 0, 0,
0, 0, 785]'),
 Text(0.828125, 0.08333333333333333, 'gini = 0.625\nsamples = 38\nvalue =
[0, 0, 0, 31, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 15, 0, 0, 0, 0, 0, 0,
0, 16]'),
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js

```
 Text(0.859375, 0.08333333333333333, 'gini = 0.365\nsamples = 623\nvalue =
[0, 0, 0, 114, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 101, 0, 0, 0, 0, 0,
0, 0, 769]'),
 Text(0.9375, 0.4166666666666667, 'x[7] <= 1.435\ngini = 0.419\nsamples = 1
116\nvalue = [0, 0, 0, 1287, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 93, 0,
0, 0, 0, 0, 0, 0, 388]'),
 Text(0.90625, 0.25, 'x[14] <= 7.435\ngini = 0.664\nsamples = 124\nvalue =
[0, 0, 0, 65, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 62, 0, 0, 0, 0, 0, 0,
0, 77]'),
 Text(0.890625, 0.08333333333333333, 'gini = 0.587\nsamples = 42\nvalue =
[0, 0, 0, 38, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 20, 0, 0, 0, 0, 0, 0,
0, 11]'),
 Text(0.921875, 0.08333333333333333, 'gini = 0.624\nsamples = 82\nvalue =
[0, 0, 0, 27, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 42, 0, 0, 0, 0, 0, 0,
0, 66]'),
 Text(0.96875, 0.25, 'x[9] <= 41.53\ngini = 0.35\nsamples = 992\nvalue =
[0, 0, 0, 1222, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 31, 0, 0, 0, 0, 0,
0, 0, 311]'),
 Text(0.953125, 0.08333333333333333, 'gini = 0.247\nsamples = 532\nvalue =
[0, 0, 0, 689, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 11, 0, 0, 0, 0, 0, 0,
0, 103]'),
 Text(0.984375, 0.08333333333333333, 'gini = 0.434\nsamples = 460\nvalue =
[0, 0, 0, 533, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 20, 0, 0, 0, 0, 0, 0,
0, 208]')]
```



concusion

The bestfit model is logistic Regression with score of 0.49849376303049575