In [1]:
```python
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]:
```python
df =pd.read_csv(r"D:\datasets\madrid_2010.csv")
df
```

Out[2]:

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

209448 rows × 17 columns

In [3]:
```python
df.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 [4]:

```
df1 =df.fillna(value=0)
df1
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY
0	2010-03-01 01:00:00	0.00	0.29	0.00	0.0	0.00	25.090000	29.219999	0.0
1	2010-03-01 01:00:00	0.00	0.27	0.00	0.0	0.00	24.879999	30.040001	0.0
2	2010-03-01 01:00:00	0.00	0.28	0.00	0.0	0.00	17.410000	20.540001	0.0
3	2010-03-01 01:00:00	0.38	0.24	1.74	0.0	0.05	15.610000	21.080000	0.0
4	2010-03-01 01:00:00	0.79	0.00	1.32	0.0	0.00	21.430000	26.070000	0.0
...	...	...	...	...	...	...	...	...	...
209443	2010-08-01 00:00:00	0.00	0.55	0.00	0.0	0.00	125.000000	219.899994	0.0
209444	2010-08-01 00:00:00	0.00	0.27	0.00	0.0	0.00	45.709999	47.410000	0.0
209445	2010-08-01 00:00:00	0.00	0.00	0.00	0.0	0.24	46.560001	49.040001	0.0
209446	2010-08-01 00:00:00	0.00	0.00	0.00	0.0	0.00	46.770000	50.119999	0.0
209447	2010-08-01 00:00:00	0.92	0.43	0.71	0.0	0.25	76.330002	88.190002	0.0

209448 rows × 17 columns

In [5]:

```
df1.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      209448 non-null  float64
 2   CO       209448 non-null  float64
 3   EBE      209448 non-null  float64
 4   MXY      209448 non-null  float64
 5   NMHC     209448 non-null  float64
 6   NO_2     209448 non-null  float64
 7   NOx      209448 non-null  float64
 8   OXY      209448 non-null  float64
 9   O_3      209448 non-null  float64
 10  PM10     209448 non-null  float64
 11  PM25     209448 non-null  float64
 12  PXY      209448 non-null  float64
 13  SO_2     209448 non-null  float64
 14  TCH      209448 non-null  float64
 15  TOL      209448 non-null  float64
 16  station  209448 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 27.2+ 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	0.29	0.00	0.0	0.00	25.090000	29.219999	0.0	68.930000
1	0.00	0.27	0.00	0.0	0.00	24.879999	30.040001	0.0	0.000000
2	0.00	0.28	0.00	0.0	0.00	17.410000	20.540001	0.0	72.120003
3	0.38	0.24	1.74	0.0	0.05	15.610000	21.080000	0.0	72.970001
4	0.79	0.00	1.32	0.0	0.00	21.430000	26.070000	0.0	0.000000
...	...	...	...	...	...	...	...	...	...
209443	0.00	0.55	0.00	0.0	0.00	125.000000	219.899994	0.0	25.379999
209444	0.00	0.27	0.00	0.0	0.00	45.709999	47.410000	0.0	0.000000
209445	0.00	0.00	0.00	0.0	0.24	46.560001	49.040001	0.0	46.250000
209446	0.00	0.00	0.00	0.0	0.00	46.770000	50.119999	0.0	77.709999
209447	0.92	0.43	0.71	0.0	0.25	76.330002	88.190002	0.0	52.259998

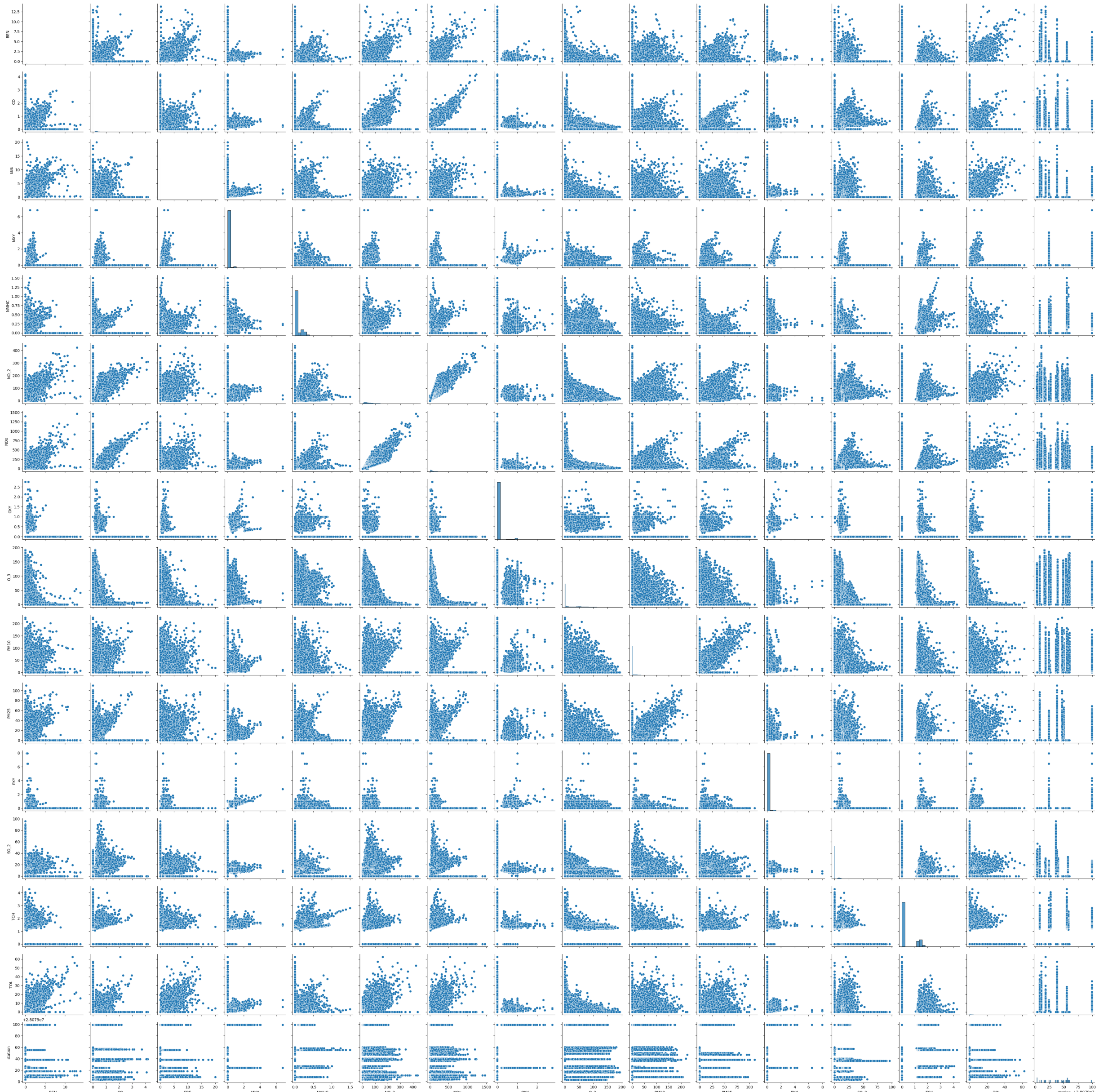
209448 rows × 16 columns

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)
```

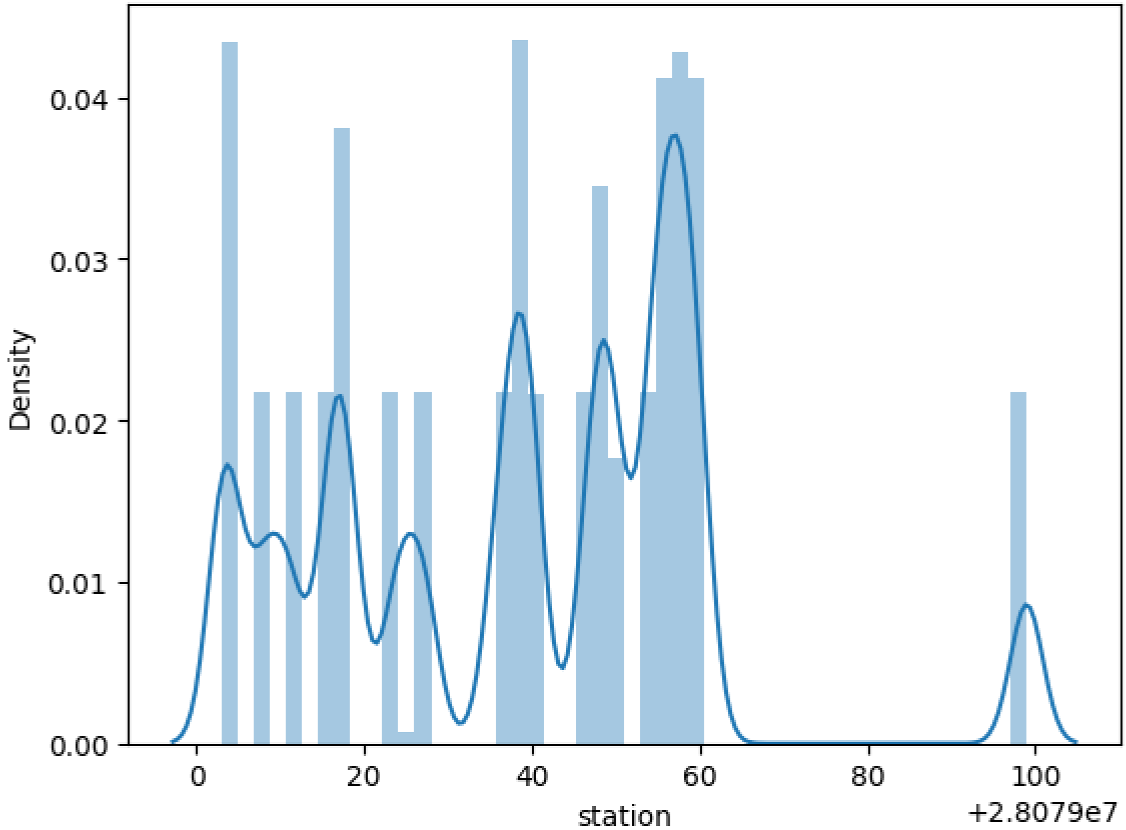
Out[8]: <seaborn.axisgrid.PairGrid at 0x224b9fb7050>



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

```
C:\Users\HP\AppData\Local\Temp\ipykernel_12480\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	-3.675069
CO	-13.393441
EBE	-5.093590
MXY	5.385968
NMHC	32.325774
NO_2	-0.076891
NOx	0.044861
OXY	15.656012
O_3	0.024323
PM10	0.285897
PM25	0.189816
PXY	3.531780
SO_2	-0.916592
TCH	1.002794
TOL	0.633842

In [15]:

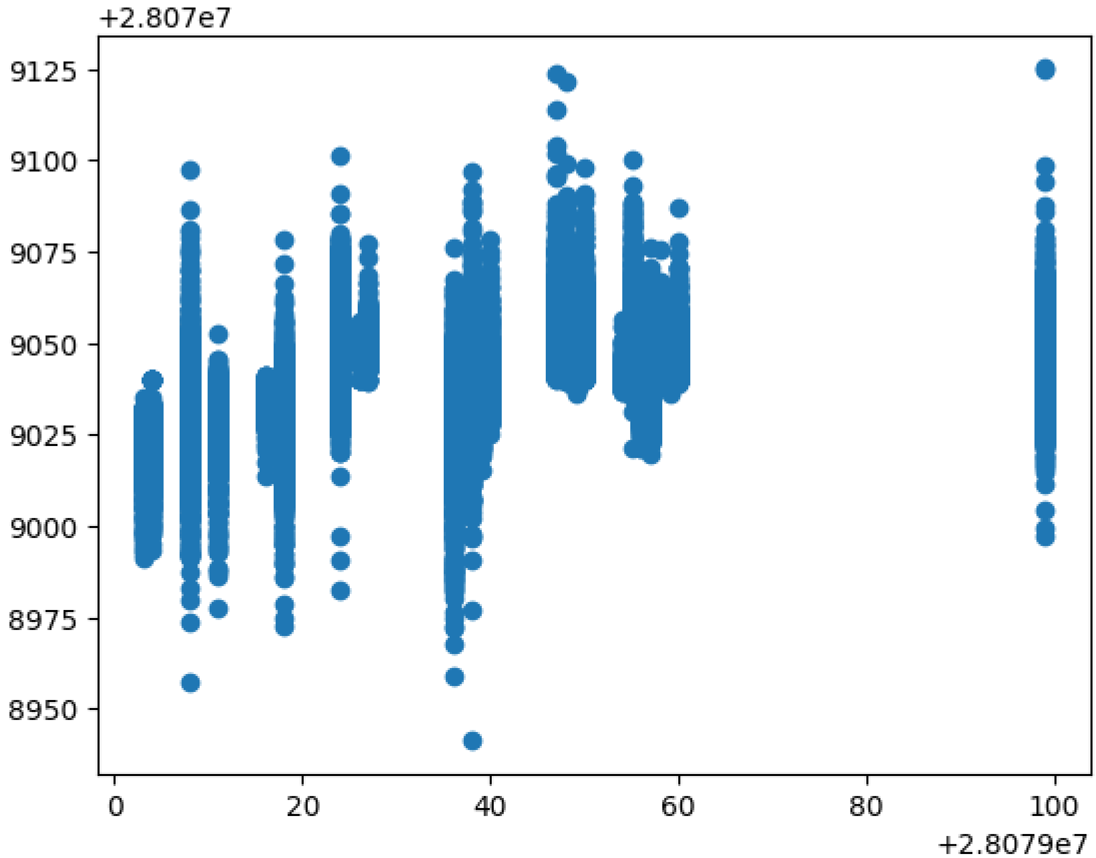
```
print(lr.intercept_)
```

```
28079039.760603778
```

In [16]:

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

Out[16]: <matplotlib.collections.PathCollection at 0x2248f7cf3d0>



In [17]:
```
print(lr.score(x_test,y_test))
```

```
0.22524695026649333
```

In [18]:
```
print(lr.score(x_train,y_train))
```

```
0.2198822349850672
```

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]:
```
0.22522514571473184
```

Loading [MathJax]/extensions/Safe.js

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]:
```
0.11260673000732624
```

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.         -1.03442475  0.20003777  0.         -0.07837882
  0.01520194  0.29743903  0.03318777  0.29556583  0.25905033  0.13878282
 -1.07099494  1.93662963 -0.65519968]
```

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

```
28079040.872002963
```

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

```
[28079027.02765574 28079035.2384622  28079043.83452355 ...
 28079045.45205421 28079027.19544104 28079031.6041041 ]
```

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

```
0.15060496110645605
```

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]:
```
(209448, 15)
```

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

Out[32]:
```
(209448,)
```

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([28079003, 28079004, 28079008, 28079011, 28079016, 28079017,
       28079018, 28079024, 28079026, 28079027, 28079036, 28079038,
       28079039, 28079040, 28079047, 28079048, 28079049, 28079050,
       28079054, 28079055, 28079056, 28079057, 28079058, 28079059,
       28079060, 28079099], dtype=int64)
```

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

Out[39]:
```
0.7488398074939842
```

In [40]:
```
logr.predict_proba(observation)[0][0]
```
Out[40]:
```
2.8940987299868526e-170
```
In [41]:
```
logr.predict_proba(observation)[0][1]
```
Out[41]:
```
3.553838813539422e-201
```
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 [48]:
```
grid_search =GridSearchCV(estimator =rfc,param_grid=parameters,cv=2,scoring=
grid_search.fit(x_train,y_train)
```
Out[48]:
```
▶ GridSearchCV
▶ estimator: RandomForestClassifier
  ▶ RandomForestClassifier
```
In [49]:
```
grid_search.best_score_
```
Out[49]:
```
0.7313397205334691
```
In [50]:
```
rfc_best=grid_search.best_estimator_
```
In [51]:
```
py.figure(figsize=(80,50))
plot_tree(rfc_best.estimators_[5],filled=True)
```

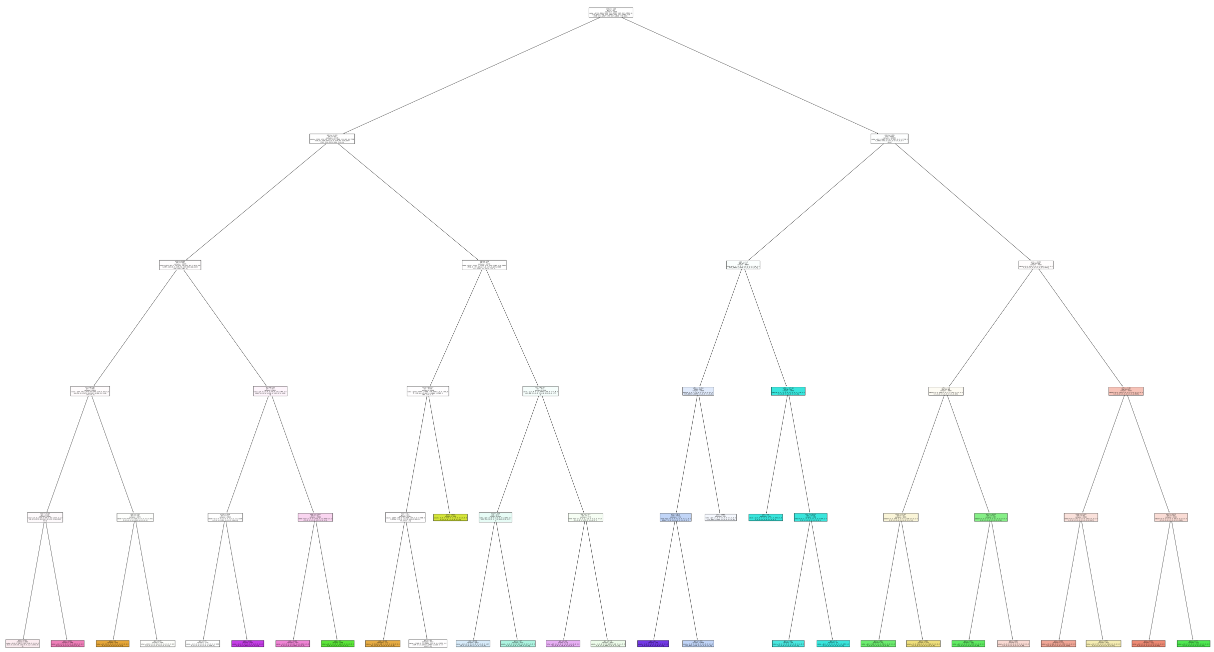
Out[51]:

```
[Text(0.5023148148148148, 0.9166666666666666, 'x[10] <= 0.28\ngini = 0.96\n
samples = 39653\nvalue = [2700, 2650, 2653, 2595, 2752, 1880, 2674, 2525, 8
0\n2598, 2625, 2749, 2658, 2645, 2647, 1503, 2546, 2016\n2616, 2554, 2537,
2490, 2529, 2603, 2318, 2691]'),
 Text(0.27314814814814814, 0.75, 'x[6] <= 32.565\ngini = 0.945\nsamples = 2
9145\nvalue = [2700, 2650, 13, 2595, 2752, 1880, 2674, 60, 80, 2598\n2625,
11, 2658, 2645, 22, 0, 2546, 42, 2616, 2554\n2537, 2490, 2529, 2603, 2318,
0]'),
 Text(0.14814814814814814, 0.5833333333333334, 'x[9] <= 0.665\ngini = 0.935
\nsamples = 10273\nvalue = [433, 482, 2, 516, 927, 795, 877, 60, 22, 610, 4
54\n6, 904, 1027, 8, 0, 1175, 42, 1167, 859, 222, 1238\n1744, 1593, 1041,
0]'),
 Text(0.07407407407407407, 0.4166666666666667, 'x[1] <= 0.06\ngini = 0.898
\nsamples = 6838\nvalue = [433, 482, 2, 516, 927, 795, 7, 60, 3, 610, 0, 4
\n904, 18, 0, 0, 1175, 38, 1167, 8, 222, 11, 1744\n1593, 35, 0]'),
 Text(0.037037037037037035, 0.25, 'x[4] <= 0.025\ngini = 0.842\nsamples = 4
966\nvalue = [0, 8, 2, 516, 3, 795, 3, 60, 3, 610, 0, 4, 2\n18, 0, 0, 1175,
38, 1167, 8, 9, 11, 1744, 1593\n35, 0]'),
 Text(0.018518518518518517, 0.08333333333333333, 'gini = 0.792\nsamples = 3
480\nvalue = [0, 8, 2, 516, 3, 795, 3, 60, 3, 1, 0, 4, 2\n18, 0, 0, 1175, 3
8, 1167, 8, 9, 11, 7, 1593, 35\n0]'),
 Text(0.05555555555555555, 0.08333333333333333, 'gini = 0.384\nsamples = 14
86\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 609, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 1737, 0, 0, 0]'),
 Text(0.1111111111111111, 0.25, 'x[8] <= 3.54\ngini = 0.756\nsamples = 1872
\nvalue = [433, 474, 0, 0, 924, 0, 4, 0, 0, 0, 0, 0, 902\n0, 0, 0, 0, 0, 0,
0, 213, 0, 0, 0, 0, 0]'),
 Text(0.09259259259259259, 0.08333333333333333, 'gini = 0.0\nsamples = 306
\nvalue = [0, 474, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0]'),
 Text(0.12962962962962962, 0.08333333333333333, 'gini = 0.69\nsamples = 156
6\nvalue = [433, 0, 0, 0, 924, 0, 4, 0, 0, 0, 0, 0, 902\n0, 0, 0, 0, 0, 0,
0, 213, 0, 0, 0, 0, 0]'),
 Text(0.2222222222222222, 0.4166666666666667, 'x[1] <= 0.03\ngini = 0.824\n
samples = 3435\nvalue = [0, 0, 0, 0, 0, 0, 870, 0, 19, 0, 454, 2, 0\n1009,
8, 0, 0, 4, 0, 851, 0, 1227, 0, 0, 1006\n0]'),
 Text(0.18518518518518517, 0.25, 'x[13] <= 0.61\ngini = 0.672\nsamples = 18
12\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 19, 0, 0, 2, 0, 1009\n8, 0, 0, 4, 0, 8
51, 0, 0, 0, 0, 1006, 0]'),
 Text(0.16666666666666666, 0.08333333333333333, 'gini = 0.51\nsamples = 127
5\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 2, 0, 1009\n8, 0, 0, 4, 0, 6,
0, 0, 0, 0, 1006, 0]'),
 Text(0.2037037037037037, 0.08333333333333333, 'gini = 0.041\nsamples = 537
\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 18, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 845,
0, 0, 0, 0, 0, 0]'),
 Text(0.25925925925925924, 0.25, 'x[14] <= 0.055\ngini = 0.621\nsamples = 1
623\nvalue = [0, 0, 0, 0, 0, 0, 870, 0, 0, 0, 454, 0, 0, 0\n0, 0, 0, 0, 0,
0, 0, 1227, 0, 0, 0, 0]'),
 Text(0.24074074074074073, 0.08333333333333333, 'gini = 0.401\nsamples = 10
64\nvalue = [0, 0, 0, 0, 0, 0, 10, 0, 0, 0, 454, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 1227, 0, 0, 0, 0]'),
 Text(0.2777777777777778, 0.08333333333333333, 'gini = 0.0\nsamples = 559\n
value = [0, 0, 0, 0, 0, 0, 860, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0]'),
 Text(0.39814814814814814, 0.5833333333333334, 'x[9] <= 0.74\ngini = 0.941
 = 18872\nvalue = [2267, 2168, 11, 2079, 1825, 1085, 1797, 0, 58,
```

```
1988\n2171, 5, 1754, 1618, 14, 0, 1371, 0, 1449, 1695\n2315, 1252, 785, 101
0, 1277, 0]'),
 Text(0.35185185185185186, 0.4166666666666667, 'x[14] <= 0.055\ngini = 0.90
9\nsamples = 12666\nvalue = [2267, 2168, 2, 2079, 1825, 1085, 2, 0, 0, 198
8, 0\n0, 1754, 11, 0, 0, 1371, 0, 1449, 1, 2315, 0\n785, 1010, 2, 0]'),
 Text(0.3333333333333333, 0.25, 'x[8] <= 0.3\ngini = 0.901\nsamples = 11403
\nvalue = [2267, 2168, 2, 36, 1825, 1085, 0, 0, 0, 1988, 0\n0, 1754, 11, 0,
0, 1371, 0, 1449, 0, 2315, 0\n785, 1010, 2, 0]'),
 Text(0.3148148148148148, 0.08333333333333333, 'gini = 0.101\nsamples = 143
1\nvalue = [7, 2168, 0, 36, 2, 32, 0, 0, 0, 18, 0, 0, 0\n11, 0, 0, 1, 0, 0,
0, 2, 0, 8, 2, 0, 0]'),
 Text(0.35185185185185186, 0.08333333333333333, 'gini = 0.89\nsamples = 997
2\nvalue = [2260, 0, 2, 0, 1823, 1053, 0, 0, 0, 1970, 0, 0\n1754, 0, 0, 0,
1370, 0, 1449, 0, 2313, 0, 777\n1008, 2, 0]'),
 Text(0.37037037037037035, 0.25, 'gini = 0.003\nsamples = 1263\nvalue = [0,
0, 0, 2043, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0]'),
 Text(0.4444444444444444, 0.4166666666666667, 'x[2] <= 0.055\ngini = 0.83\n
samples = 6206\nvalue = [0, 0, 9, 0, 0, 0, 1795, 0, 58, 0, 2171, 5, 0\n160
7, 14, 0, 0, 0, 0, 1694, 0, 1252, 0, 0, 1275\n0]'),
 Text(0.4074074074074074, 0.25, 'x[1] <= 0.03\ngini = 0.74\nsamples = 4027
\nvalue = [0, 0, 0, 0, 0, 0, 20, 0, 0, 0, 2171, 0, 0\n1607, 14, 0, 0, 0, 0,
17, 0, 1252, 0, 0, 1275\n0]'),
 Text(0.3888888888888889, 0.08333333333333333, 'gini = 0.504\nsamples = 182
3\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1607\n14, 0, 0, 0, 0, 1
7, 0, 1, 0, 0, 1275, 0]'),
 Text(0.42592592592592593, 0.08333333333333333, 'gini = 0.47\nsamples = 220
4\nvalue = [0, 0, 0, 0, 0, 0, 20, 0, 0, 0, 2171, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 1251, 0, 0, 0, 0]'),
 Text(0.48148148148148145, 0.25, 'x[9] <= 5.575\ngini = 0.52\nsamples = 217
9\nvalue = [0, 0, 9, 0, 0, 0, 1775, 0, 58, 0, 0, 5, 0, 0\n0, 0, 0, 0, 0, 16
77, 0, 0, 0, 0, 0, 0]'),
 Text(0.46296296296296297, 0.08333333333333333, 'gini = 0.508\nsamples = 21
5\nvalue = [0, 0, 0, 0, 0, 0, 110, 0, 14, 0, 0, 1, 0, 0\n0, 0, 0, 0, 0, 19
6, 0, 0, 0, 0, 0, 0]'),
 Text(0.5, 0.08333333333333333, 'gini = 0.516\nsamples = 1964\nvalue = [0,
0, 9, 0, 0, 0, 1665, 0, 44, 0, 0, 4, 0, 0\n0, 0, 0, 0, 0, 1481, 0, 0, 0, 0,
0, 0]'),
 Text(0.7314814814814815, 0.75, 'x[1] <= 0.055\ngini = 0.852\nsamples = 105
08\nvalue = [0, 0, 2640, 0, 0, 0, 0, 2465, 0, 0, 0, 2738, 0\n0, 2625, 1503,
0, 1974, 0, 0, 0, 0, 0, 0, 0\n2691]'),
 Text(0.6111111111111112, 0.5833333333333334, 'x[12] <= 3.0\ngini = 0.737\n
samples = 5592\nvalue = [0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 2738, 0, 0\n2625,
1503, 0, 1974, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(0.5740740740740741, 0.4166666666666667, 'x[5] <= 33.015\ngini = 0.65
\nsamples = 3919\nvalue = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0\n2625,
1503, 0, 1974, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(0.5555555555555556, 0.25, 'x[5] <= 5.14\ngini = 0.631\nsamples = 1438
\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0\n1095, 571, 0, 569, 0,
0, 0, 0, 0, 0, 0, 0]'),
 Text(0.5370370370370371, 0.08333333333333333, 'gini = 0.0\nsamples = 15\nv
alue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 25, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(0.5740740740740741, 0.08333333333333333, 'gini = 0.628\nsamples = 142
3\nvalue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0\n1095, 571, 0, 544, 0,
0, 0, 0, 0, 0, 0, 0]'),
```

```
 Text(0.5925925925925926, 0.25, 'gini = 0.654\nsamples = 2481\nvalue = [0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0\n1530, 932, 0, 1405, 0, 0, 0, 0, 0,
0, 0, 0]'),
 Text(0.6481481481481481, 0.4166666666666667, 'x[6] <= 127.15\ngini = 0.001
\nsamples = 1673\nvalue = [0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 2734, 0, 0\n0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(0.6296296296296297, 0.25, 'gini = 0.0\nsamples = 1364\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 2250, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(0.6666666666666666, 0.25, 'x[5] <= 72.695\ngini = 0.008\nsamples = 30
9\nvalue = [0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 484, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0]'),
 Text(0.6481481481481481, 0.08333333333333333, 'gini = 0.142\nsamples = 18
\nvalue = [0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 24, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0]'),
 Text(0.6851851851851852, 0.08333333333333333, 'gini = 0.0\nsamples = 291\n
value = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 460, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0]'),
 Text(0.8518518518518519, 0.5833333333333334, 'x[4] <= 0.155\ngini = 0.666
\nsamples = 4916\nvalue = [0, 0, 2637, 0, 0, 0, 0, 2465, 0, 0, 0, 0, 0\n0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2691]'),
 Text(0.7777777777777778, 0.4166666666666667, 'x[7] <= 0.1\ngini = 0.548\ns
amples = 1882\nvalue = [0, 0, 1434, 0, 0, 0, 0, 1346, 0, 0, 0, 0, 0\n0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 156]'),
 Text(0.7407407407407407, 0.25, 'x[14] <= 1.385\ngini = 0.52\nsamples = 169
5\nvalue = [0, 0, 1434, 0, 0, 0, 0, 1134, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 74]'),
 Text(0.7222222222222222, 0.08333333333333333, 'gini = 0.345\nsamples = 602
\nvalue = [0, 0, 166, 0, 0, 0, 0, 745, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 33]'),
 Text(0.7592592592592593, 0.08333333333333333, 'gini = 0.389\nsamples = 109
3\nvalue = [0, 0, 1268, 0, 0, 0, 0, 389, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 41]'),
 Text(0.8148148148148148, 0.25, 'x[0] <= 0.595\ngini = 0.402\nsamples = 187
\nvalue = [0, 0, 0, 0, 0, 0, 0, 212, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 82]'),
 Text(0.7962962962962963, 0.08333333333333333, 'gini = 0.272\nsamples = 136
\nvalue = [0, 0, 0, 0, 0, 0, 0, 175, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 34]'),
 Text(0.8333333333333334, 0.08333333333333333, 'gini = 0.492\nsamples = 51
\nvalue = [0, 0, 0, 0, 0, 0, 0, 37, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 48]'),
 Text(0.9259259259259259, 0.4166666666666667, 'x[7] <= 0.15\ngini = 0.613\n
samples = 3034\nvalue = [0, 0, 1203, 0, 0, 0, 0, 1119, 0, 0, 0, 0, 0\n0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 2535]'),
 Text(0.8888888888888888, 0.25, 'x[12] <= 9.575\ngini = 0.585\nsamples = 19
32\nvalue = [0, 0, 1203, 0, 0, 0, 0, 339, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1547]'),
 Text(0.8703703703703703, 0.08333333333333333, 'gini = 0.498\nsamples = 113
1\nvalue = [0, 0, 468, 0, 0, 0, 0, 156, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1175]'),
 Text(0.9074074074074074, 0.08333333333333333, 'gini = 0.572\nsamples = 801
\nvalue = [0, 0, 735, 0, 0, 0, 0, 183, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 372]'),
 Text(0.9629629629629629, 0.25, 'x[4] <= 0.275\ngini = 0.493\nsamples = 110
3\nvalue = [0, 0, 0, 0, 0, 0, 0, 780, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 0, 988]'),
 Text(0.9444444444444444, 0.08333333333333333, 'gini = 0.352\nsamples = 751
\nvalue = [0, 0, 0, 0, 0, 0, 0, 275, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 933]'),
 Text(0.9814814814814815, 0.08333333333333333, 'gini = 0.177\nsamples = 351
\nvalue = [0, 0, 0, 0, 0, 0, 0, 505, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 55]')]
```



concusion

The bestfit model is logistic Regression with score of 0.7488398074939842

In [ ]: