In [1]:
```python
# import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```python
data=pd.read_csv(r"C:\Users\user\Desktop\DINESH\C10_air\madrid_2008.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PI
0	2008-06-01 01:00:00	NaN	0.47	NaN	NaN	NaN	83.089996	120.699997	NaN	16.990000	16.889
1	2008-06-01 01:00:00	NaN	0.59	NaN	NaN	NaN	94.820000	130.399994	NaN	17.469999	19.040
2	2008-06-01 01:00:00	NaN	0.55	NaN	NaN	NaN	75.919998	104.599998	NaN	13.470000	20.270
3	2008-06-01 01:00:00	NaN	0.36	NaN	NaN	NaN	61.029999	66.559998	NaN	23.110001	10.850
4	2008-06-01 01:00:00	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61	12.120000	37.160
...	...	...	...	...	...	...	...	...	...	...	
226387	2008-11-01 00:00:00	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91	57.400002	5.450
226388	2008-11-01 00:00:00	NaN	0.30	NaN	NaN	NaN	41.880001	48.500000	NaN	35.830002	15.020
226389	2008-11-01 00:00:00	0.25	NaN	0.56	NaN	0.11	83.610001	102.199997	NaN	14.130000	17.540
226390	2008-11-01 00:00:00	0.54	NaN	2.70	NaN	0.18	70.639999	81.860001	NaN	NaN	11.910
226391	2008-11-01 00:00:00	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64	31.910000	12.690

226392 rows × 17 columns

In [3]:
```
data.head(10)
```

Out[3]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	2008-06-01 01:00:00	NaN	0.47	NaN	NaN	NaN	83.089996	120.699997	NaN	16.990000	16.889999
1	2008-06-01 01:00:00	NaN	0.59	NaN	NaN	NaN	94.820000	130.399994	NaN	17.469999	19.040001
2	2008-06-01 01:00:00	NaN	0.55	NaN	NaN	NaN	75.919998	104.599998	NaN	13.470000	20.270000
3	2008-06-01 01:00:00	NaN	0.36	NaN	NaN	NaN	61.029999	66.559998	NaN	23.110001	10.850000
4	2008-06-01 01:00:00	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61	12.120000	37.160000
5	2008-06-01 01:00:00	NaN	0.47	NaN	NaN	0.22	67.820000	101.099998	NaN	20.610001	23.389999
6	2008-06-01 01:00:00	0.17	0.40	0.44	NaN	0.15	72.639999	91.220001	NaN	17.040001	19.940001
7	2008-06-01 01:00:00	NaN	0.51	NaN	NaN	NaN	80.440002	141.500000	NaN	10.310000	37.259998
8	2008-06-01 01:00:00	NaN	0.36	NaN	NaN	NaN	68.150002	85.639999	NaN	23.580000	15.060000
9	2008-06-01 01:00:00	NaN	0.18	NaN	NaN	0.16	58.330002	64.769997	NaN	35.060001	7.400000

In [4]:
```
data.tail(20)
```

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM
226372	2008-11-01 00:00:00	NaN	0.46	NaN	NaN	0.15	73.089996	101.800003	NaN	23.990000	17.3700
226373	2008-11-01 00:00:00	0.18	0.35	0.13	NaN	0.17	56.099998	75.300003	NaN	31.480000	14.6800
226374	2008-11-01 00:00:00	NaN	0.35	NaN	NaN	NaN	61.529999	71.570000	NaN	22.190001	14.8300
226375	2008-11-01 00:00:00	NaN	0.27	NaN	NaN	NaN	51.759998	63.180000	NaN	29.889999	8.3700
226376	2008-11-01 00:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
226377	2008-11-01 00:00:00	NaN	0.53	NaN	NaN	NaN	72.949997	109.500000	NaN	22.170000	17.6299
226378	2008-11-01 00:00:00	NaN	0.51	NaN	NaN	NaN	44.799999	54.709999	NaN	33.490002	13.0700
226379	2008-11-01 00:00:00	NaN	0.24	NaN	NaN	NaN	55.950001	62.240002	NaN	38.970001	9.0400
226380	2008-11-01 00:00:00	NaN	0.37	NaN	NaN	NaN	57.070000	68.129997	NaN	30.280001	7.8900
226381	2008-11-01 00:00:00	NaN	0.28	NaN	NaN	NaN	40.090000	45.380001	NaN	43.130001	12.2200
226382	2008-11-01 00:00:00	NaN	0.29	NaN	NaN	NaN	38.540001	45.730000	NaN	47.180000	7.9600
226383	2008-11-01 00:00:00	NaN	0.30	NaN	NaN	NaN	56.650002	58.189999	NaN	30.250000	14.3000
226384	2008-11-01 00:00:00	NaN	0.52	NaN	NaN	NaN	78.019997	101.400002	NaN	30.540001	8.4500
226385	2008-11-01 00:00:00	NaN	0.18	NaN	NaN	NaN	41.389999	48.389999	NaN	38.980000	9.7800
226386	2008-11-01 00:00:00	1.19	0.35	1.03	NaN	0.04	71.419998	83.040001	NaN	17.930000	N
226387	2008-11-01 00:00:00	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91	57.400002	5.4500
226388	2008-11-01 00:00:00	NaN	0.30	NaN	NaN	NaN	41.880001	48.500000	NaN	35.830002	15.0200

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM
226389	2008-11-01 00:00:00	0.25	NaN	0.56	NaN	0.11	83.610001	102.199997	NaN	14.130000	17.5400
226390	2008-11-01 00:00:00	0.54	NaN	2.70	NaN	0.18	70.639999	81.860001	NaN	NaN	11.9100
226391	2008-11-01 00:00:00	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64	31.910000	12.6900

In [5]:
```
data.describe()
```

Out[5]:

	BEN	CO	EBE	MXY	NMHC	NO_2	
count	67047.000000	208109.000000	67044.000000	25867.000000	85079.000000	225315.000000	2:
mean	0.892794	0.412052	1.273285	2.444665	0.192408	55.357186	
std	1.149072	0.303645	1.288372	2.402643	0.125238	33.990306	
min	0.000000	0.000000	0.100000	0.240000	0.000000	0.000000	
25%	0.200000	0.240000	0.620000	1.000000	0.110000	30.245000	
50%	0.460000	0.330000	1.000000	1.620000	0.160000	49.669998	
75%	1.090000	0.490000	1.400000	3.100000	0.240000	73.610001	
max	32.340000	8.280000	47.270000	55.889999	2.880000	554.900024	

In [6]:
```
np.shape(data)
```

Out[6]: (226392, 17)

In [7]:
```
np.size(data)
```

Out[7]: 3848664

In [8]:
```
data.isna()
```

Out[8]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10	PM25	PXY
0	False	True	False	True	True	True	False	False	True	False	False	False	True
1	False	True	False	True	True	True	False	False	True	False	False	True	True
2	False	True	False	True	True	True	False	False	True	False	False	True	True
3	False	True	False	True	True	True	False	False	True	False	False	True	True
4	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
226387	False	False	False	False	False	False	False	False	False	False	False	False	False
226388	False	True	False	True	True	True	False	False	True	False	False	True	True
226389	False	False	True	False	True	False	False	False	True	False	False	False	True
226390	False	False	True	False	True	False	False	False	True	True	False	True	True
226391	False	False	False	False	False	False	False	False	False	False	False	False	False

226392 rows × 17 columns

In [9]:
```
data.dropna()
```

Out[9]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PI
4	2008-06-01 01:00:00	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994	1.61	12.120000	37.160
21	2008-06-01 01:00:00	0.32	0.37	1.00	0.39	0.33	21.580000	22.180000	1.00	35.770000	7.900
25	2008-06-01 01:00:00	0.73	0.39	1.04	1.70	0.18	64.839996	86.709999	1.31	23.379999	14.760
30	2008-06-01 02:00:00	1.95	0.51	1.98	3.77	0.24	79.750000	143.399994	2.03	18.090000	31.139
47	2008-06-01 02:00:00	0.36	0.39	0.39	0.50	0.34	26.790001	27.389999	1.00	33.029999	7.620
...	...	...	...	...	...	...	...	...	...	...	
226362	2008-10-31 23:00:00	0.47	0.35	0.65	1.00	0.33	22.480000	25.020000	1.00	33.509998	10.200
226366	2008-10-31 23:00:00	0.92	0.46	1.21	2.75	0.19	78.440002	106.199997	1.70	18.320000	14.140
226371	2008-11-01 00:00:00	1.83	0.53	2.22	4.51	0.17	93.260002	158.399994	2.38	18.770000	20.750
226387	2008-11-01 00:00:00	0.48	0.30	0.57	1.00	0.31	13.050000	14.160000	0.91	57.400002	5.450
226391	2008-11-01 00:00:00	0.75	0.36	1.20	2.75	0.16	58.240002	74.239998	1.64	31.910000	12.690

25631 rows × 17 columns

In [10]:
```
data.columns
```

Out[10]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_
3',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [11]:
```
sd=data[['BEN','CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx']]
```

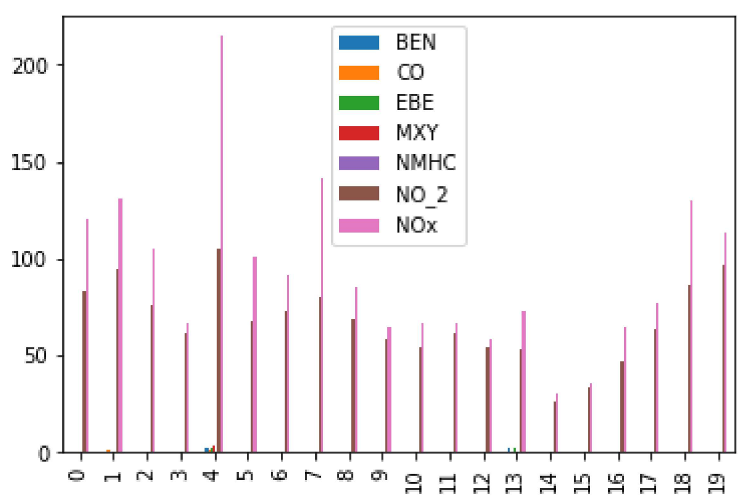
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx
0	NaN	0.47	NaN	NaN	NaN	83.089996	120.699997
1	NaN	0.59	NaN	NaN	NaN	94.820000	130.399994
2	NaN	0.55	NaN	NaN	NaN	75.919998	104.599998
3	NaN	0.36	NaN	NaN	NaN	61.029999	66.559998
4	1.68	0.80	1.70	3.01	0.30	105.199997	214.899994
5	NaN	0.47	NaN	NaN	0.22	67.820000	101.099998
6	0.17	0.40	0.44	NaN	0.15	72.639999	91.220001
7	NaN	0.51	NaN	NaN	NaN	80.440002	141.500000
8	NaN	0.36	NaN	NaN	NaN	68.150002	85.639999
9	NaN	0.18	NaN	NaN	0.16	58.330002	64.769997
10	NaN	0.45	NaN	NaN	NaN	53.700001	66.610001
11	NaN	0.15	NaN	NaN	NaN	60.849998	66.900002
12	NaN	0.26	NaN	NaN	NaN	53.700001	58.240002
13	2.20	0.36	1.88	NaN	0.08	52.529999	72.709999
14	NaN	0.23	NaN	NaN	NaN	26.430000	30.620001
15	NaN	0.27	NaN	NaN	NaN	33.310001	35.169998
16	NaN	0.37	NaN	NaN	NaN	47.240002	64.139999
17	NaN	0.39	NaN	NaN	NaN	63.360001	76.800003
18	NaN	0.45	NaN	NaN	NaN	86.150002	130.100006
19	NaN	0.30	NaN	NaN	NaN	96.519997	113.199997

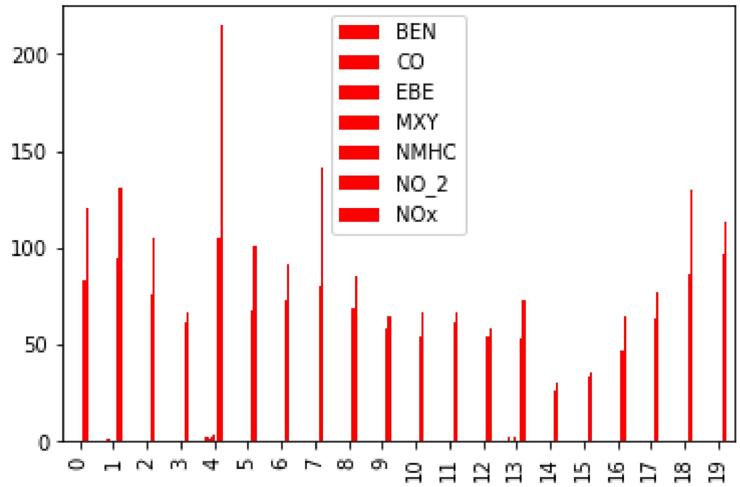
In [13]:
```
dd.plot.bar()
```

Out[13]: <AxesSubplot:>



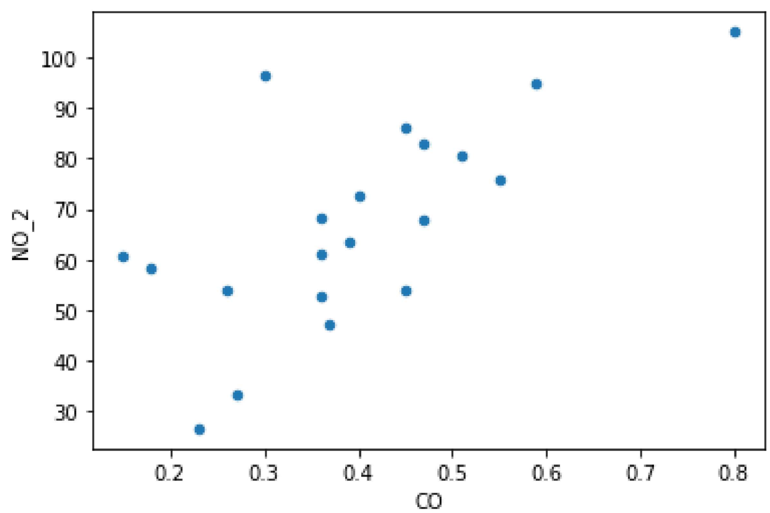
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



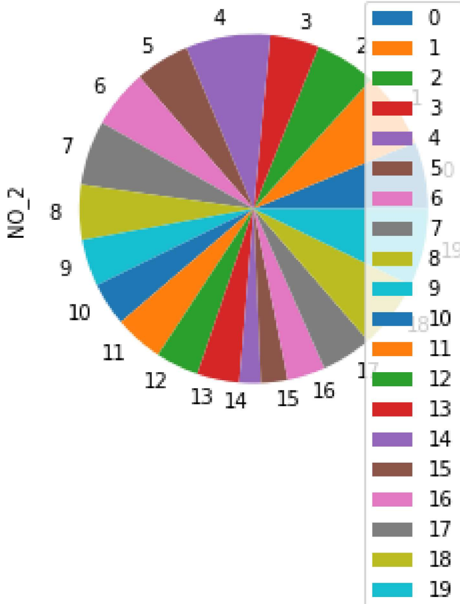
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

Out[15]: <AxesSubplot:xlabel='CO', ylabel='NO_2'>



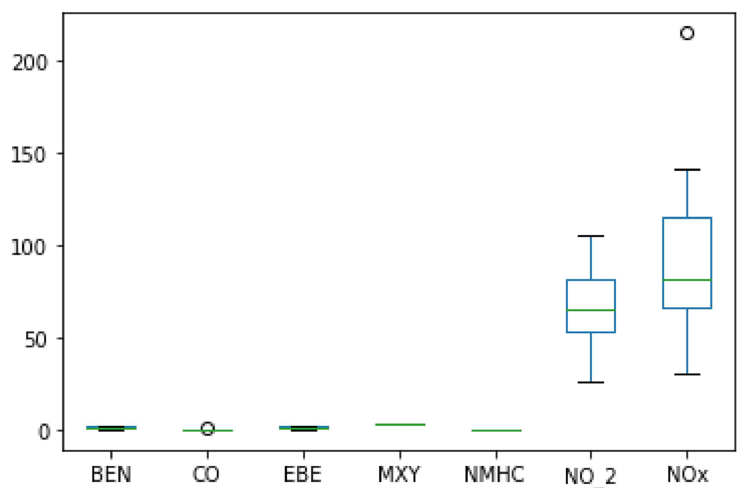
```
In [16]: dd.plot.pie(y='NO_2')
```

Out[16]: <AxesSubplot:ylabel='NO_2'>



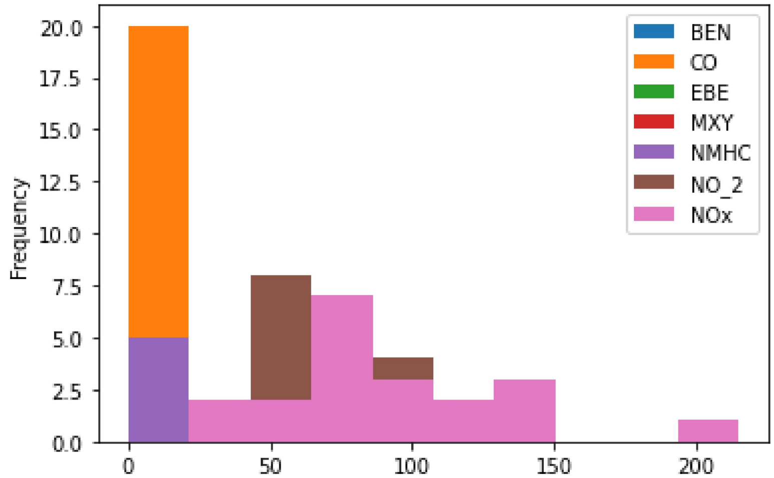

```
In [17]: dd.plot.box()
```

Out[17]: <AxesSubplot:>



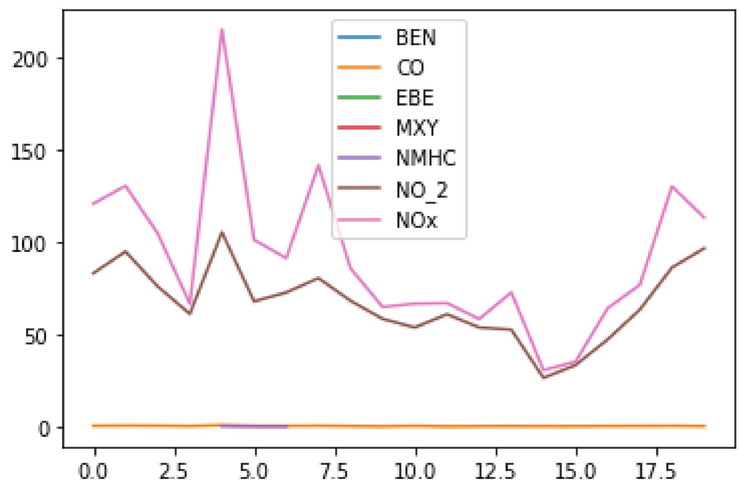
In [18]:
```
dd.plot.hist()
```

Out[18]: <AxesSubplot:ylabel='Frequency'>

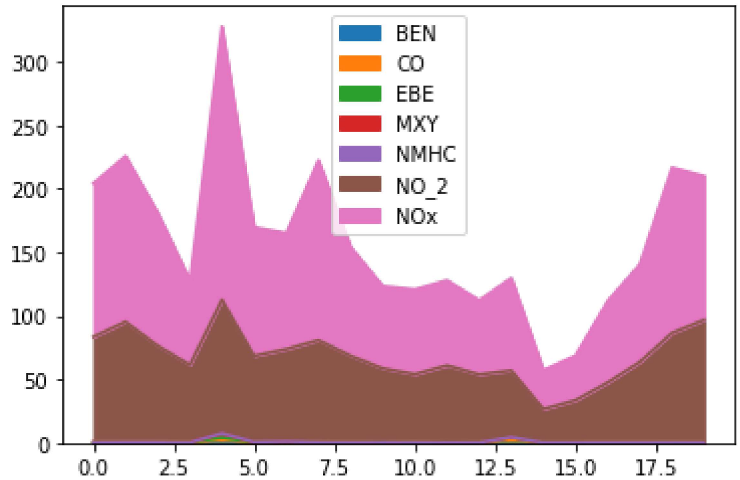


In [19]:
```
dd.plot.line()
```

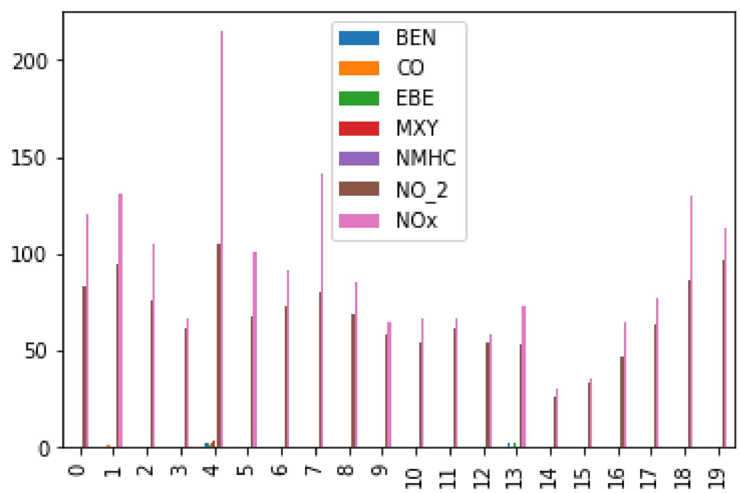
Out[19]: <AxesSubplot:>



In [20]:
```
dd.plot.area()
```
Out[20]: <AxesSubplot:>

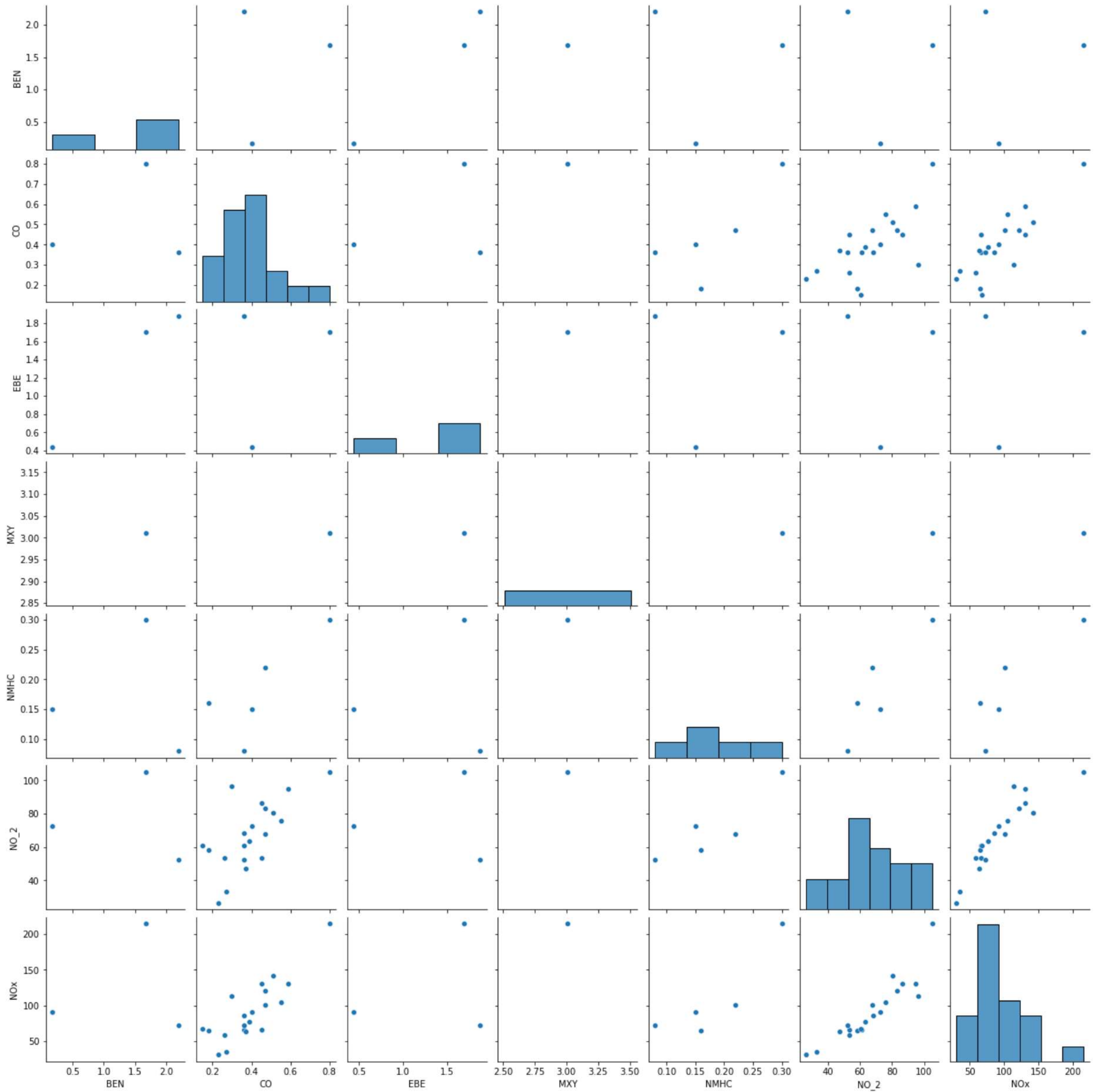


In [21]:
```
dd.plot.bar()
```
Out[21]: <AxesSubplot:>



In [22]:
```
sns.pairplot(dd)
```

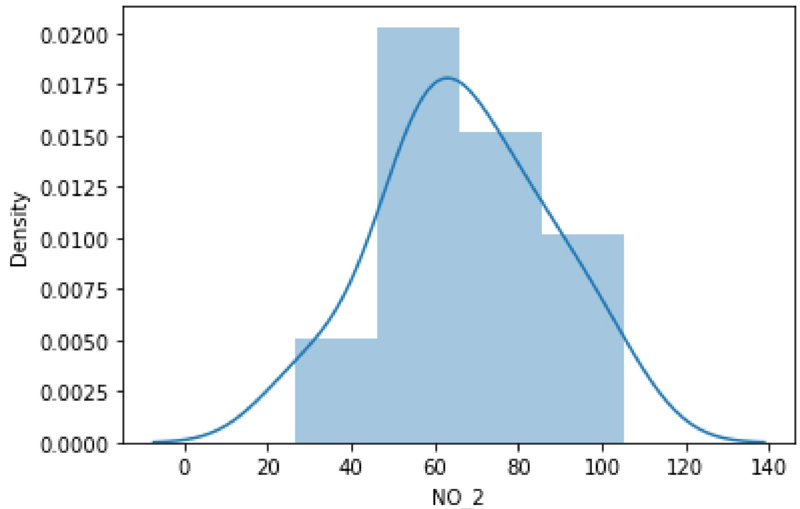
Out[22]: <seaborn.axisgrid.PairGrid at 0x23a05e31040>



In [23]:
```
sns.distplot(dd['NO_2'])
```

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

Out[23]: <AxesSubplot:xlabel='NO_2', ylabel='Density'>



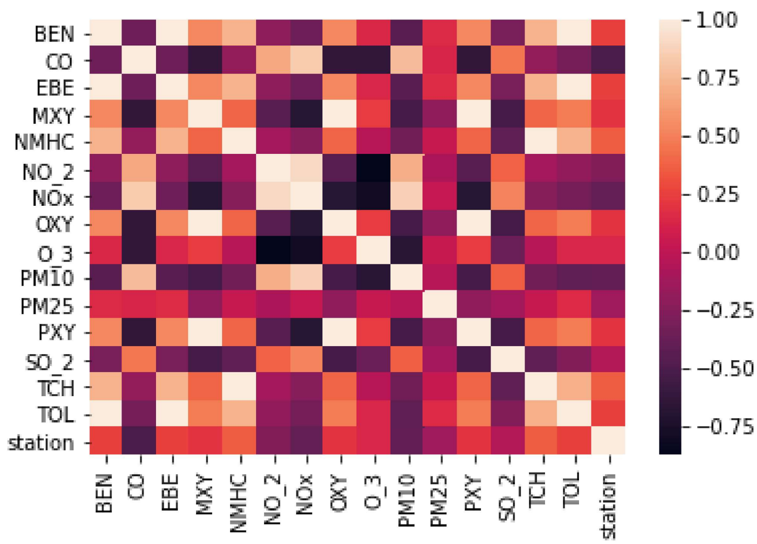
In [24]:
```
ds=data.fillna(20)
```

In [25]:
```
ssd=ds.head(20)
```

In [26]:
```
sd1=ssd[['BEN','CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx']]
```

In [27]:
```
sns.heatmap(ssd.corr())
```

Out[27]: <AxesSubplot:>



In [28]:
```
x= ssd[['BEN','CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx']]
y=ssd['station']
```

In [29]:
```
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 [30]:
```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[30]:
```
LinearRegression()
```

In [31]:
```
print(lr.intercept_)
```

```
28079074.373621885
```

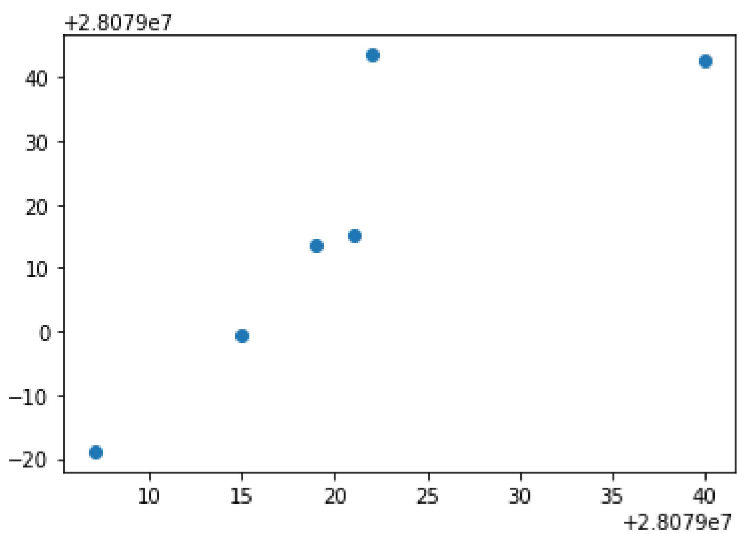
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	-0.309065
CO	-81.630467
EBE	-0.346497
MXY	-3.083410
NMHC	1.401695
NO_2	0.893791
NOx	-0.405432

In [33]:
```
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[33]:
```
<matplotlib.collections.PathCollection at 0x23a0cf172e0>
```



In [34]:
```
print(lr.score(x_test,y_test))
```

```
-1.4078693822476107
```

In [35]:
```
lr.score(x_test,y_test)
```

Out[35]:
```
-1.4078693822476107
```

In [36]:
```
lr.score(x_train,y_train)
```

Out[36]:
```
0.444354312677673
```

In [37]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [38]:
```
dr=Ridge(alpha=10)
dr.fit(x_train,y_train)
```

Out[38]:
```
Ridge(alpha=10)
```

In [39]:
```
dr.score(x_test,y_test)
```

Out[39]:
```
0.07917877945677099
```

In [40]:
```
dr.score(x_train,y_train)
```

Out[40]:
```
0.34958770559594676
```

In [41]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[41]:
```
Lasso(alpha=10)
```

In [42]:
```
la.score(x_test,y_test)
```

Out[42]:
```
-0.11091526836025944
```

In [43]:
```
la.score(x_train,y_train)
```

Out[43]:
```
0.19409297505634304
```

ElasticNet

In [44]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[44]:
```
ElasticNet()
```

In [45]:
```
print(en.coef_)
```

```
[-0.00396767 -0.          -0.03054865 -1.95581725  0.54991684  0.55571409
 -0.4475284 ]
```

In [46]:
```
print(en.intercept_)
```

```
28079048.275279436
```

In [47]:
```
prediction=en.predict(x_test)
```

In [48]:
```
print(en.score(x_test,y_test))
```

```
0.09055436471620404
```

In [49]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [50]:
```
from sklearn.linear_model import LogisticRegression
```

In [51]:
```
feature_matrix = ssd[['BEN','CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx']]
target_vector=ssd['station']
```

In [52]:
```
feature_matrix.shape
```

Out[52]:
```
(20, 7)
```

In [53]:
```
target_vector.shape
```

Out[53]:
```
(20,)
```

In [54]:
```
from sklearn.preprocessing import StandardScaler
```

In [55]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [56]:
```
logr= LogisticRegression()
logr.fit(fs,target_vector)
```

Out[56]:
```
LogisticRegression()
```

In [57]:
```
observation =[[1.2,2.3,3.3,4.3,5.3,6.3,7.3]]
```

In [58]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
[28079003]
```

In [59]:
```
logr.classes_
```
Out[59]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079018, 28079019, 28079021, 28079022, 28079036, 28079038,
       28079039, 28079040], dtype=int64)
```
In [60]:
```
logr.predict_proba(observation)[0][0]
```
Out[60]:
```
0.03454457148367317
```
In [61]:
```
ged=data[['BEN','CO','EBE','MXY','NMHC','NO_2','NOx','OXY','O_3','PM10','PXY','
```
In [62]:
```
d=ged.fillna(20)
```
In [63]:
```
dg=d.head(100)
```
In [64]:
```
x=dg[['BEN','CO','EBE','MXY','NMHC','NO_2','NOx','OXY','O_3','PM10','PXY','SO_2
y=dg['station']
```
In [65]:
```
print(len(x))
print(len(y))
```
```
100
100
```
In [66]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```
In [67]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```
Out[67]:
```
RandomForestClassifier()
```
In [68]:
```
paramets = {'max_depth':[1,2,3,4,5,6,7],
              'min_samples_leaf':[5,10,15,20,25,30,35],
              'n_estimators':[10,20,30,40,50,60,70]}
```

In [69]:
```
from sklearn.model_selection import GridSearchCV
grid_search= GridSearchCV(estimator = rfc,param_grid=paramets,cv=2,scoring="acc
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.py:
666: UserWarning: The least populated class in y has only 1 members, which is
less than n_splits=2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[69]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5, 6, 7],
                         'min_samples_leaf': [5, 10, 15, 20, 25, 30, 35],
                         'n_estimators': [10, 20, 30, 40, 50, 60, 70]},
             scoring='accuracy')
```

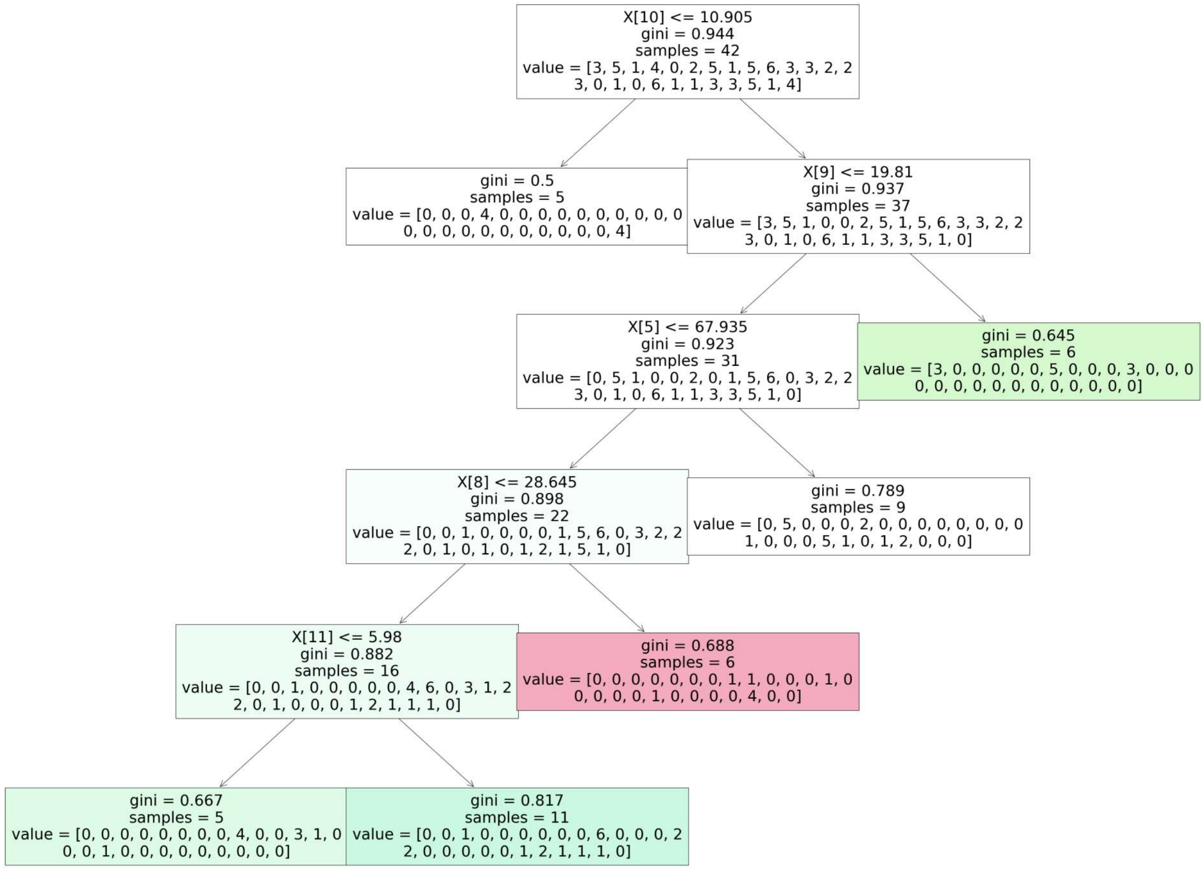
In [70]:
```
grid_search.best_score_
```

Out[70]:
```
0.35714285714285715
```

In [71]:
```
rfc_best=grid_search.best_estimator_
```

In [72]:
```python
from sklearn.tree import plot_tree
plt.figure(figsize=(50,40))
plot_tree(rfc_best.estimators_[5],filled=True)
```

Out[72]:
```
[Text(1594.2857142857142, 1993.2, 'X[10] <= 10.905\ngini = 0.944\nsamples = 4
2\nvalue = [3, 5, 1, 4, 0, 2, 5, 1, 5, 6, 3, 3, 2, 2\n3, 0, 1, 0, 6, 1, 1, 3,
3, 5, 1, 4]'),
 Text(1195.7142857142858, 1630.8000000000002, 'gini = 0.5\nsamples = 5\nvalue
= [0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
4]'),
 Text(1992.8571428571427, 1630.8000000000002, 'X[9] <= 19.81\ngini = 0.937\ns
amples = 37\nvalue = [3, 5, 1, 0, 0, 2, 5, 1, 5, 6, 3, 3, 2, 2\n3, 0, 1, 0,
6, 1, 1, 3, 3, 5, 1, 0]'),
 Text(1594.2857142857142, 1268.4, 'X[5] <= 67.935\ngini = 0.923\nsamples = 31
\nvalue = [0, 5, 1, 0, 0, 2, 0, 1, 5, 6, 0, 3, 2, 2\n3, 0, 1, 0, 6, 1, 1, 3,
3, 5, 1, 0]'),
 Text(1195.7142857142858, 906.0, 'X[8] <= 28.645\ngini = 0.898\nsamples = 22
\nvalue = [0, 0, 1, 0, 0, 0, 0, 1, 5, 6, 0, 3, 2, 2\n2, 0, 1, 0, 1, 0, 1, 2,
1, 5, 1, 0]'),
 Text(797.1428571428571, 543.5999999999999, 'X[11] <= 5.98\ngini = 0.882\nsam
ples = 16\nvalue = [0, 0, 1, 0, 0, 0, 0, 0, 4, 6, 0, 3, 1, 2\n2, 0, 1, 0, 0,
0, 1, 2, 1, 1, 1, 0]'),
 Text(398.57142857142856, 181.19999999999982, 'gini = 0.667\nsamples = 5\nval
ue = [0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 3, 1, 0\n0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0]'),
 Text(1195.7142857142858, 181.19999999999982, 'gini = 0.817\nsamples = 11\nva
lue = [0, 0, 1, 0, 0, 0, 0, 0, 0, 6, 0, 0, 0, 2\n2, 0, 0, 0, 0, 0, 1, 2, 1,
1, 1, 0]'),
 Text(1594.2857142857142, 543.5999999999999, 'gini = 0.688\nsamples = 6\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0\n0, 0, 0, 0, 1, 0, 0, 0, 0, 4,
0, 0]'),
 Text(1992.8571428571427, 906.0, 'gini = 0.789\nsamples = 9\nvalue = [0, 5,
0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0\n1, 0, 0, 0, 5, 1, 0, 1, 2, 0, 0, 0]'),
 Text(2391.4285714285716, 1268.4, 'gini = 0.645\nsamples = 6\nvalue = [3, 0,
0, 0, 0, 0, 5, 0, 0, 0, 3, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]')]
```



Conclusion : ElasticNet() 0.09055436471620404 HIGH RANGE

In [ ]:

In [ ]: