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_2006.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PI
0	2006-02-01 01:00:00	NaN	1.84	NaN	NaN	NaN	155.100006	490.100006	NaN	4.880000	97.570
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.36	0.32	94.339996	229.699997	3.04	7.100000	25.820
2	2006-02-01 01:00:00	NaN	1.25	NaN	NaN	NaN	66.800003	192.000000	NaN	4.430000	34.419
3	2006-02-01 01:00:00	NaN	1.68	NaN	NaN	NaN	103.000000	407.799988	NaN	4.830000	28.260
4	2006-02-01 01:00:00	NaN	1.31	NaN	NaN	NaN	105.400002	269.200012	NaN	6.990000	54.180
...	...	...	...	...	...	...	...	...	...	...	
230563	2006-05-01 00:00:00	5.88	0.83	6.23	NaN	0.20	112.500000	218.000000	NaN	24.389999	93.120
230564	2006-05-01 00:00:00	0.76	0.32	0.48	1.09	0.08	51.900002	54.820000	0.61	48.410000	29.469
230565	2006-05-01 00:00:00	0.96	NaN	0.69	NaN	0.19	135.100006	179.199997	NaN	11.460000	64.680
230566	2006-05-01 00:00:00	0.50	NaN	0.67	NaN	0.10	82.599998	105.599998	NaN	NaN	94.360
230567	2006-05-01 00:00:00	1.95	0.74	1.99	4.00	0.24	107.300003	160.199997	2.01	17.730000	52.490

230568 rows × 17 columns

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

Out[3]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	2006-02-01 01:00:00	NaN	1.84	NaN	NaN	NaN	155.100006	490.100006	NaN	4.88	97.570000
1	2006-02-01 01:00:00	1.68	1.01	2.38	6.360000	0.32	94.339996	229.699997	3.04	7.10	25.820000
2	2006-02-01 01:00:00	NaN	1.25	NaN	NaN	NaN	66.800003	192.000000	NaN	4.43	34.419998
3	2006-02-01 01:00:00	NaN	1.68	NaN	NaN	NaN	103.000000	407.799988	NaN	4.83	28.260000
4	2006-02-01 01:00:00	NaN	1.31	NaN	NaN	NaN	105.400002	269.200012	NaN	6.99	54.180000
5	2006-02-01 01:00:00	9.41	1.69	9.98	19.959999	0.44	142.199997	453.500000	11.31	5.99	89.190002
6	2006-02-01 01:00:00	NaN	1.28	NaN	NaN	0.57	94.320000	294.000000	NaN	6.77	55.130001
7	2006-02-01 01:00:00	0.27	1.51	0.28	NaN	0.46	144.699997	385.299988	NaN	5.30	80.150002
8	2006-02-01 01:00:00	NaN	2.65	NaN	NaN	NaN	197.100006	673.099976	NaN	2.64	142.500000
9	2006-02-01 01:00:00	NaN	1.30	NaN	NaN	NaN	130.899994	282.000000	NaN	5.14	49.029999

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

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PI
230548	2006-05-01 00:00:00	NaN	0.49	NaN	NaN	0.34	66.760002	79.610001	NaN	22.760000	35.730
230549	2006-05-01 00:00:00	0.94	0.72	1.54	NaN	0.35	139.300003	207.899994	NaN	9.960000	48.820
230550	2006-05-01 00:00:00	NaN	1.20	NaN	NaN	NaN	162.600006	271.299988	NaN	14.150000	83.309
230551	2006-05-01 00:00:00	NaN	0.92	NaN	NaN	NaN	116.599998	165.399994	NaN	17.410000	40.369
230552	2006-05-01 00:00:00	NaN	0.84	NaN	NaN	0.35	89.599998	128.300003	NaN	19.100000	47.000
230553	2006-05-01 00:00:00	NaN	0.53	NaN	NaN	NaN	56.740002	59.200001	NaN	28.719999	53.400
230554	2006-05-01 00:00:00	NaN	0.85	NaN	NaN	NaN	94.750000	166.000000	NaN	15.840000	56.090
230555	2006-05-01 00:00:00	NaN	0.70	NaN	NaN	NaN	97.629997	148.800003	NaN	13.510000	48.849
230556	2006-05-01 00:00:00	1.33	0.79	1.53	NaN	0.28	112.400002	201.399994	NaN	10.860000	75.430
230557	2006-05-01 00:00:00	NaN	0.49	NaN	NaN	NaN	96.349998	150.399994	NaN	22.299999	39.389
230558	2006-05-01 00:00:00	NaN	0.73	NaN	NaN	NaN	92.019997	103.000000	NaN	18.860001	40.439
230559	2006-05-01 00:00:00	NaN	0.55	NaN	NaN	NaN	129.300003	188.300003	NaN	14.120000	40.910
230560	2006-05-01 00:00:00	NaN	0.88	NaN	NaN	NaN	121.199997	157.600006	NaN	24.510000	50.070
230561	2006-05-01 00:00:00	NaN	0.43	NaN	NaN	NaN	60.189999	68.529999	NaN	32.779999	23.219
230562	2006-05-01 00:00:00	NaN	0.84	NaN	NaN	NaN	102.400002	184.199997	NaN	6.340000	57.910
230563	2006-05-01 00:00:00	5.88	0.83	6.23	NaN	0.20	112.500000	218.000000	NaN	24.389999	93.120
230564	2006-05-01 00:00:00	0.76	0.32	0.48	1.09	0.08	51.900002	54.820000	0.61	48.410000	29.469

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PI
230565	2006-05-01 00:00:00	0.96	NaN	0.69	NaN	0.19	135.100006	179.199997	NaN	11.460000	64.68C
230566	2006-05-01 00:00:00	0.50	NaN	0.67	NaN	0.10	82.599998	105.599998	NaN	NaN	94.36C
230567	2006-05-01 00:00:00	1.95	0.74	1.99	4.00	0.24	107.300003	160.199997	2.01	17.730000	52.49C

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

Out[5]:

	BEN	CO	EBE	MXY	NMHC	NO_2	
count	73979.000000	211665.000000	73948.000000	33422.000000	90829.000000	228855.000000	2:
mean	0.918488	0.576077	1.389325	3.766834	0.191565	60.600809	
std	1.283239	0.411184	1.895449	3.919799	0.147894	37.828635	
min	0.100000	0.000000	0.100000	0.150000	0.000000	0.570000	
25%	0.200000	0.320000	0.520000	1.190000	0.090000	32.770000	
50%	0.470000	0.480000	1.000000	2.540000	0.160000	54.000000	
75%	1.120000	0.710000	1.500000	4.910000	0.250000	80.830002	
max	45.430000	8.920000	70.940002	66.900002	3.530000	526.000000	

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

Out[6]: (230568, 17)

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

Out[7]: 3919656

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	False	False	False	False	False	False	False	False	False	False	True	False
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	True	False	True	True	True	False	False	True	False	False	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...
230563	False	False	False	False	True	False	False	False	True	False	False	True	True
230564	False	False	False	False	False	False	False	False	False	False	False	False	False
230565	False	False	True	False	True	False	False	False	True	False	False	False	True
230566	False	False	True	False	True	False	False	False	True	True	False	True	True
230567	False	False	False	False	False	False	False	False	False	False	False	False	False

230568 rows × 17 columns

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

Out[9]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
5	2006-02-01 01:00:00	9.41	1.69	9.98	19.959999	0.44	142.199997	453.500000	11.31	5.990000
22	2006-02-01 01:00:00	1.69	0.79	1.24	2.670000	0.17	59.910000	120.199997	1.11	2.450000
25	2006-02-01 01:00:00	2.35	1.47	2.64	9.660000	0.40	117.699997	346.399994	5.15	4.780000
31	2006-02-01 02:00:00	4.39	0.85	7.92	17.139999	0.25	92.059998	237.000000	9.24	5.920000
48	2006-02-01 02:00:00	1.93	0.79	1.24	2.740000	0.16	60.189999	125.099998	1.11	2.280000
...	...	...	...	...	...	...	...	...	...	...
230538	2006-04-30 23:00:00	0.42	0.40	0.37	0.430000	0.10	49.259998	51.689999	1.00	64.599998
230541	2006-04-30 23:00:00	1.63	0.94	1.53	2.200000	0.33	63.220001	211.399994	1.35	17.670000
230547	2006-05-01 00:00:00	3.99	1.06	3.71	7.960000	0.26	202.399994	343.500000	3.92	11.130000
230564	2006-05-01 00:00:00	0.76	0.32	0.48	1.090000	0.08	51.900002	54.820000	0.61	48.410000
230567	2006-05-01 00:00:00	1.95	0.74	1.99	4.000000	0.24	107.300003	160.199997	2.01	17.730000

24758 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']]
```

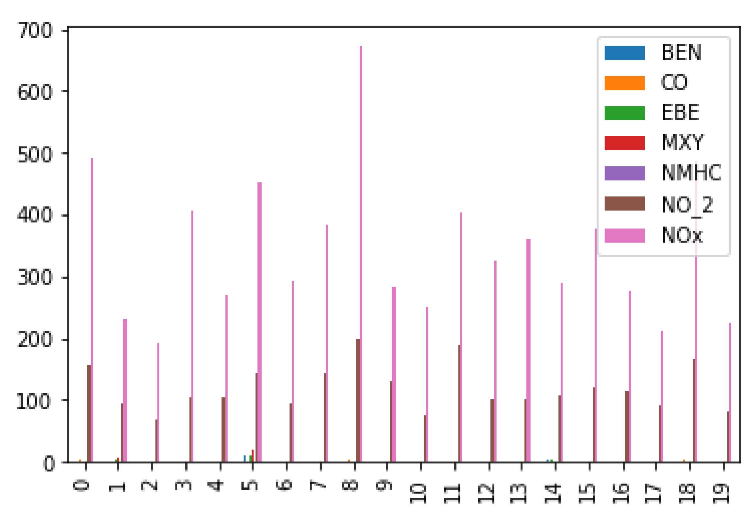
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx
0	NaN	1.84	NaN	NaN	NaN	155.100006	490.100006
1	1.68	1.01	2.38	6.360000	0.32	94.339996	229.699997
2	NaN	1.25	NaN	NaN	NaN	66.800003	192.000000
3	NaN	1.68	NaN	NaN	NaN	103.000000	407.799988
4	NaN	1.31	NaN	NaN	NaN	105.400002	269.200012
5	9.41	1.69	9.98	19.959999	0.44	142.199997	453.500000
6	NaN	1.28	NaN	NaN	0.57	94.320000	294.000000
7	0.27	1.51	0.28	NaN	0.46	144.699997	385.299988
8	NaN	2.65	NaN	NaN	NaN	197.100006	673.099976
9	NaN	1.30	NaN	NaN	NaN	130.899994	282.000000
10	NaN	1.48	NaN	NaN	0.50	75.260002	248.899994
11	NaN	1.41	NaN	NaN	NaN	189.699997	402.299988
12	NaN	1.40	NaN	NaN	NaN	100.599998	326.799988
13	NaN	1.46	NaN	NaN	NaN	102.000000	360.299988
14	2.16	1.11	2.64	NaN	0.30	105.800003	287.899994
15	NaN	1.36	NaN	NaN	NaN	121.300003	378.200012
16	NaN	1.66	NaN	NaN	NaN	113.699997	277.500000
17	NaN	0.85	NaN	NaN	NaN	89.820000	211.500000
18	NaN	1.85	NaN	NaN	NaN	165.300003	487.399994
19	NaN	1.32	NaN	NaN	NaN	82.029999	224.500000

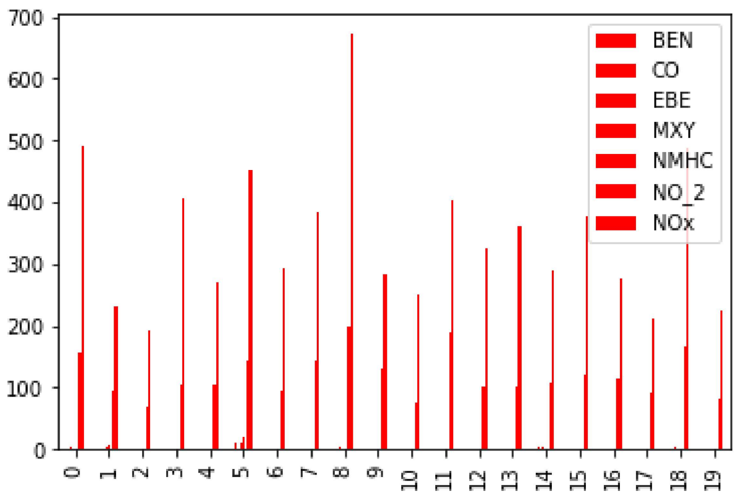
In [13]:
```
dd.plot.bar()
```

Out[13]: <AxesSubplot:>



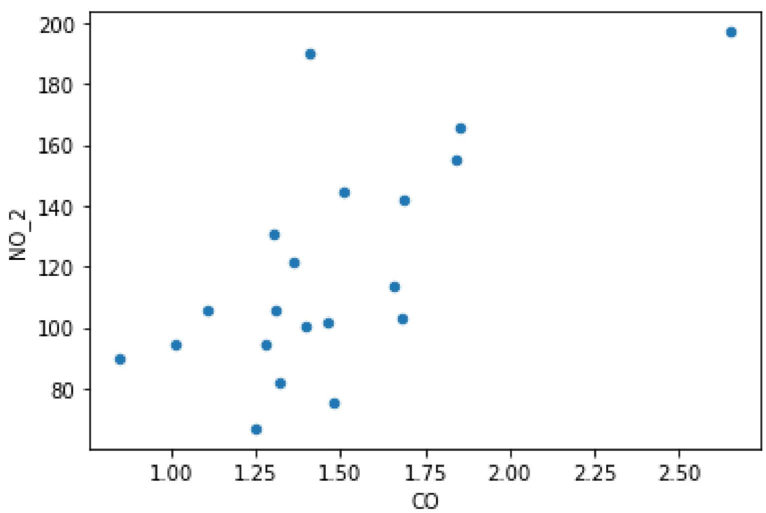
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



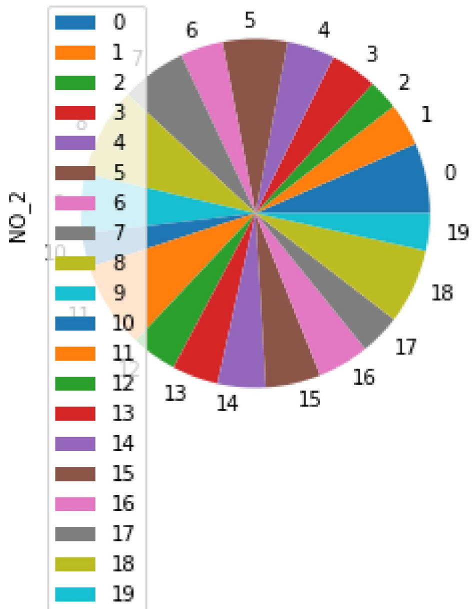
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

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



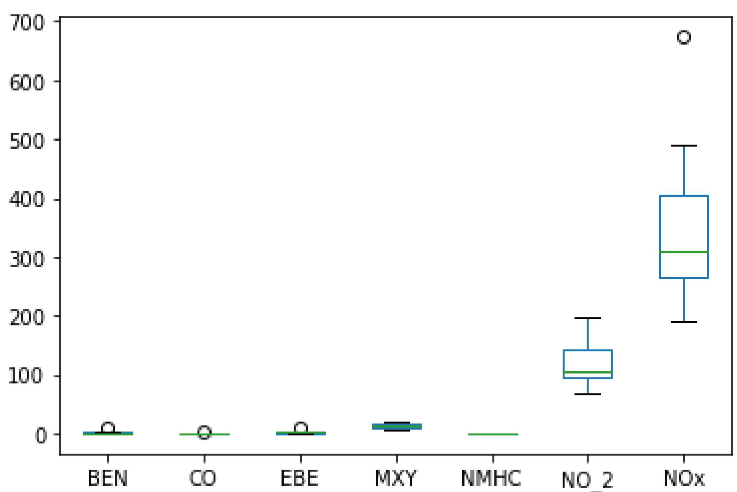
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



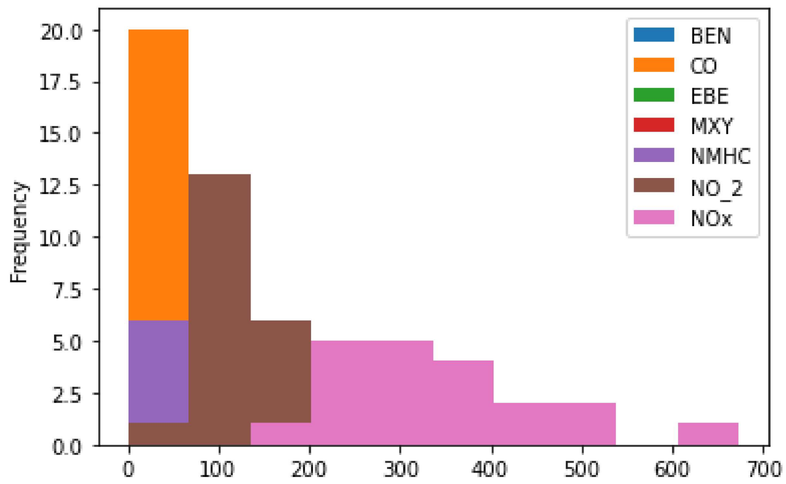
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



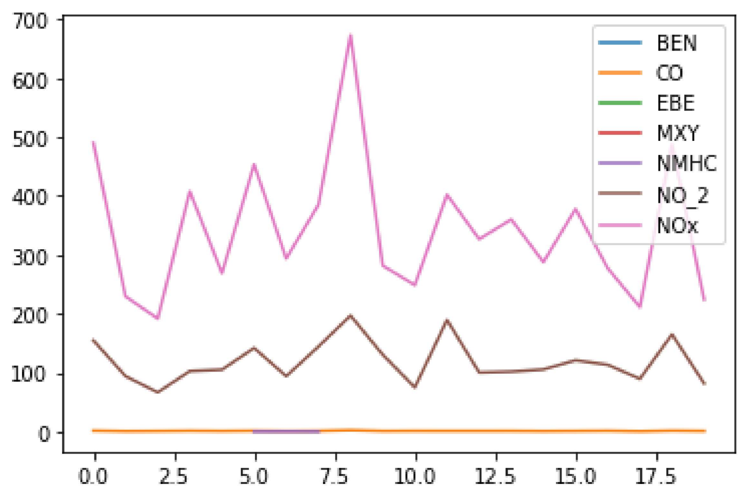
In [18]:
```
dd.plot.hist()
```

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



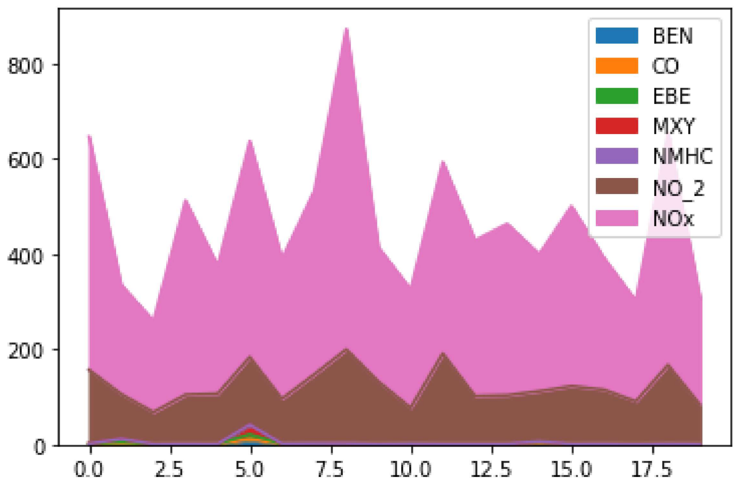
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



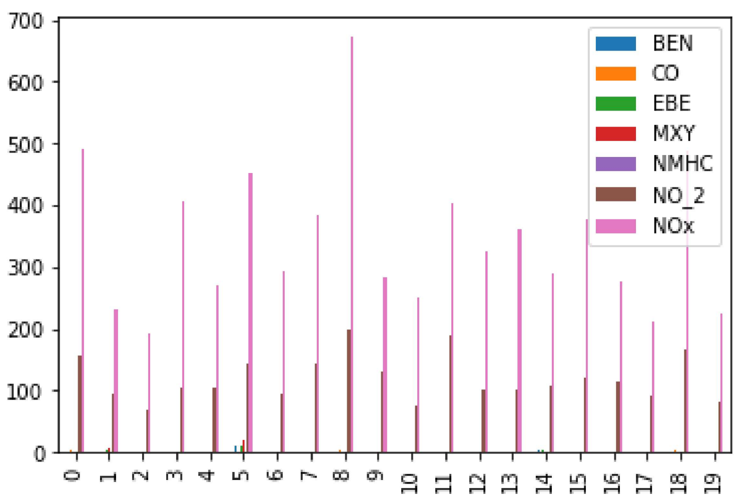
```
In [20]: dd.plot.area()
```

Out[20]: <AxesSubplot:>



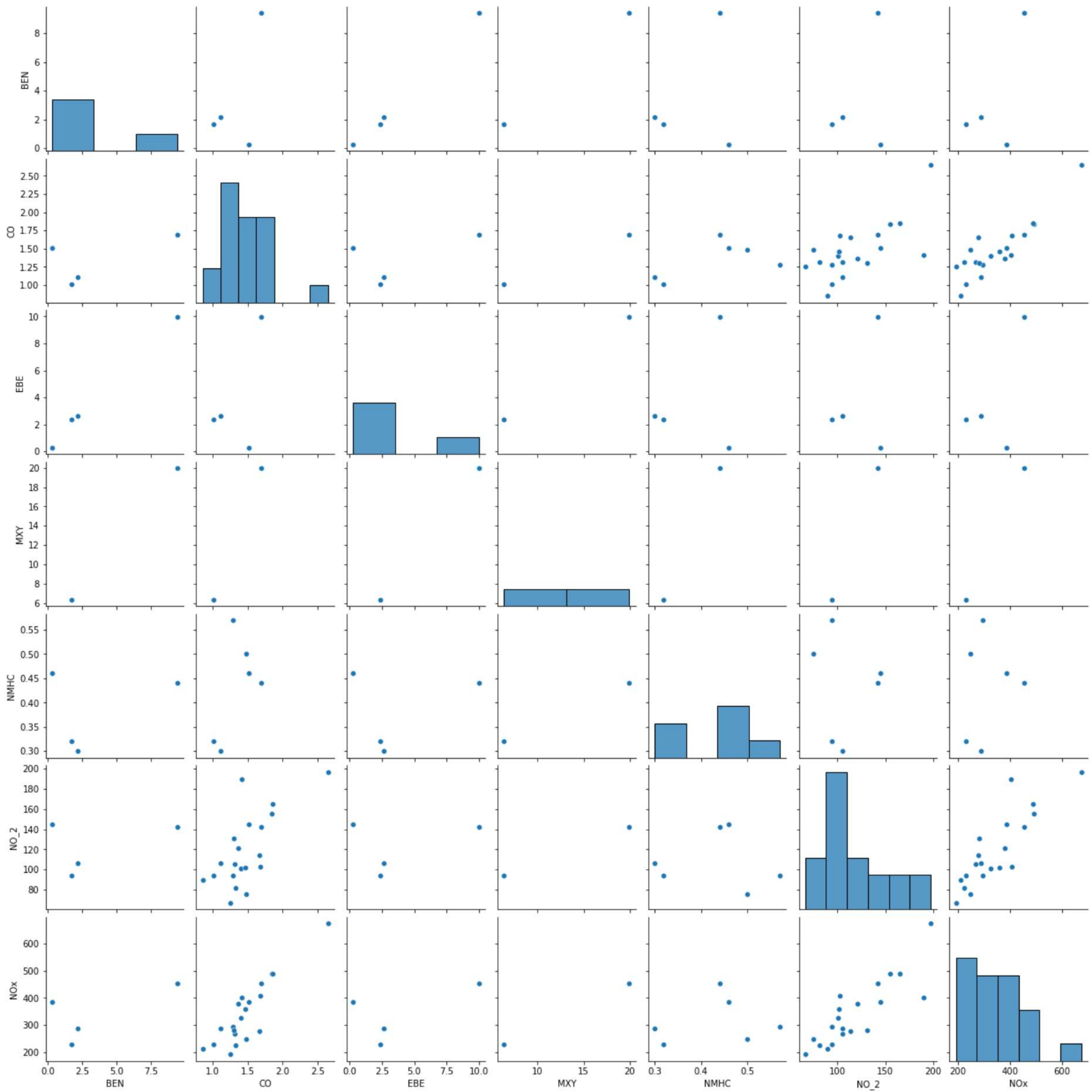

```
In [21]: dd.plot.bar()
```

Out[21]: <AxesSubplot:>



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

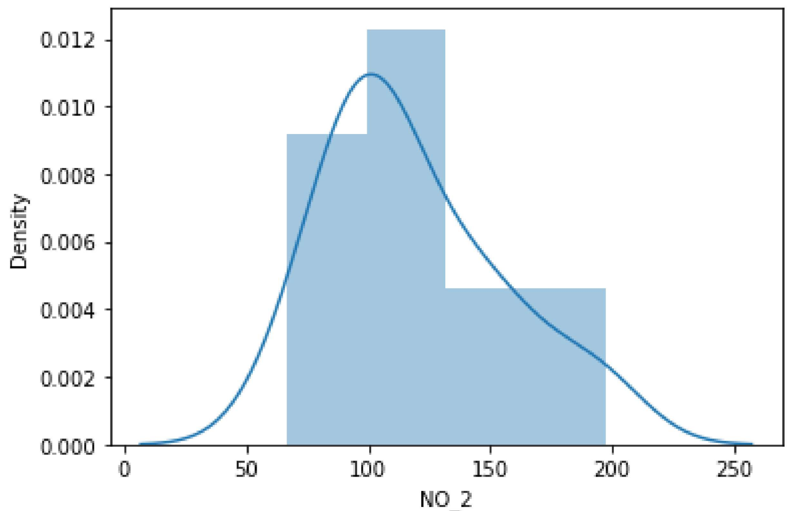
Out[22]: <seaborn.axisgrid.PairGrid at 0x1fd077e5370>



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'>
```



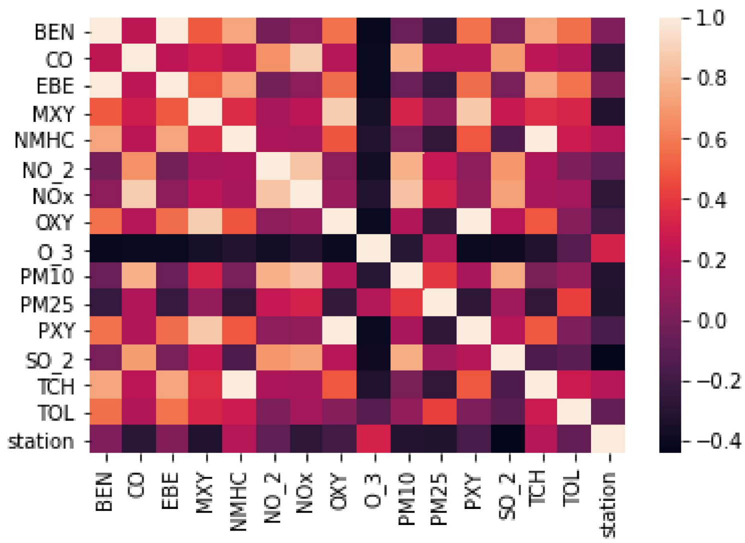
In [24]:
```
ds=data.fillna(20)
```

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_)
```

```
28078958.65525758
```

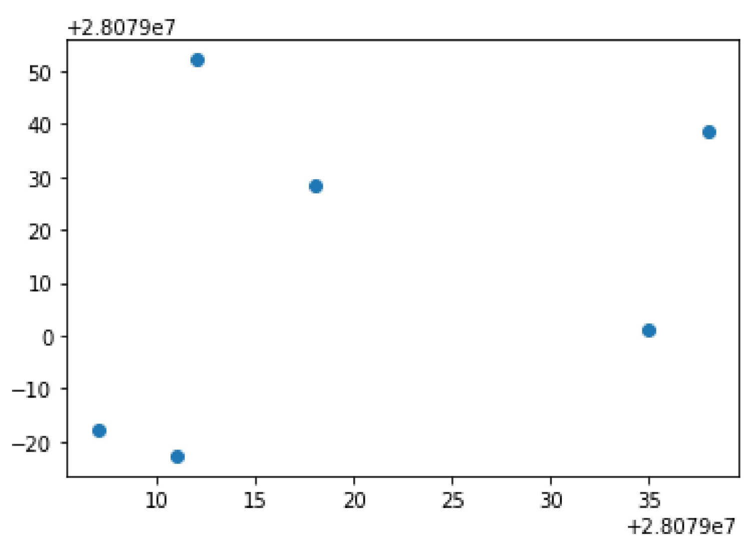
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	-29.171156
CO	-5.131412
EBE	28.505864
MXY	1.684677
NMHC	1.828117
NO_2	0.527494
NOx	-0.139827

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

Out[33]:
```
<matplotlib.collections.PathCollection at 0x1fd0e9da130>
```



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

```
-4.323431728686542
```

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

Out[35]:
```
-4.323431728686542
```

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

Out[36]:
```
0.3306654633277014
```

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]:
```
-1.4102806708085578
```

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

Out[40]:
```
0.30215538527416785
```

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]:
```
-0.33194794859185217
```

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

Out[43]:
```
0.26660301130400976
```

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_)
```

```
[ 0.17810316 -0.          0.44366156  0.          0.2013933   0.49179139
 -0.1490344 ]
```

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

```
28078998.730426773
```

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

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

```
-1.3470095664773796
```

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)
```

```
[28079009]
```

In [59]:
```
logr.classes_
```

Out[59]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079018, 28079019, 28079021, 28079035, 28079036, 28079038,
       28079039, 28079040], dtype=int64)
```

In [60]:
```
logr.predict_proba(observation)[0][0]
```

Out[60]:
```
0.07712065231736706
```

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')
```

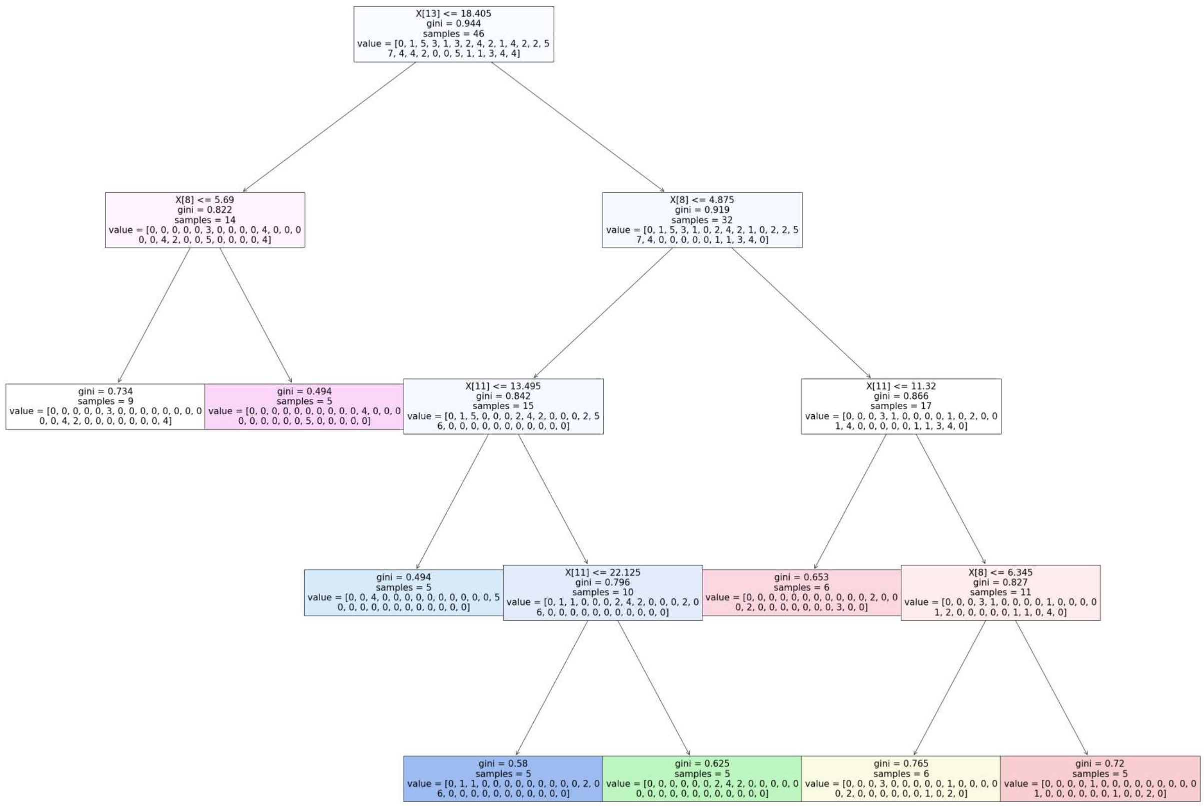
In [70]:
```
grid_search.best_score_
```

Out[70]:
```
0.3857142857142857
```

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(1046.25, 1956.96, 'X[13] <= 18.405\ngini = 0.944\nsamples = 46\nvalue =
[0, 1, 5, 3, 1, 3, 2, 4, 2, 1, 4, 2, 2, 5\n7, 4, 4, 2, 0, 0, 5, 1, 1, 3, 4,
4]'),
 Text(465.0, 1522.0800000000002, 'X[8] <= 5.69\ngini = 0.822\nsamples = 14\nv
alue = [0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 4, 0, 0, 0\n0, 0, 4, 2, 0, 0, 5, 0, 0,
0, 0, 4]'),
 Text(232.5, 1087.2, 'gini = 0.734\nsamples = 9\nvalue = [0, 0, 0, 0, 0, 3,
0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 4, 2, 0, 0, 0, 0, 0, 0, 0, 4]'),
 Text(697.5, 1087.2, 'gini = 0.494\nsamples = 5\nvalue = [0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 4, 0, 0, 0\n0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0]'),
 Text(1627.5, 1522.0800000000002, 'X[8] <= 4.875\ngini = 0.919\nsamples = 32
\nvalue = [0, 1, 5, 3, 1, 0, 2, 4, 2, 1, 0, 2, 2, 5\n7, 4, 0, 0, 0, 0, 0, 1,
1, 3, 4, 0]'),
 Text(1162.5, 1087.2, 'X[11] <= 13.495\ngini = 0.842\nsamples = 15\nvalue =
[0, 1, 5, 0, 0, 0, 2, 4, 2, 0, 0, 0, 2, 5\n6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(930.0, 652.3200000000002, 'gini = 0.494\nsamples = 5\nvalue = [0, 0, 4,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 5\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1395.0, 652.3200000000002, 'X[11] <= 22.125\ngini = 0.796\nsamples = 10
\nvalue = [0, 1, 1, 0, 0, 0, 2, 4, 2, 0, 0, 0, 2, 0\n6, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0]'),
 Text(1162.5, 217.44000000000005, 'gini = 0.58\nsamples = 5\nvalue = [0, 1,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0\n6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1627.5, 217.44000000000005, 'gini = 0.625\nsamples = 5\nvalue = [0, 0,
0, 0, 0, 0, 2, 4, 2, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2092.5, 1087.2, 'X[11] <= 11.32\ngini = 0.866\nsamples = 17\nvalue =
[0, 0, 0, 3, 1, 0, 0, 0, 0, 1, 0, 2, 0, 0\n1, 4, 0, 0, 0, 0, 0, 1, 1, 3, 4,
0]'),
 Text(1860.0, 652.3200000000002, 'gini = 0.653\nsamples = 6\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0\n0, 2, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0]'),
 Text(2325.0, 652.3200000000002, 'X[8] <= 6.345\ngini = 0.827\nsamples = 11\n
value = [0, 0, 0, 3, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0\n1, 2, 0, 0, 0, 0, 0, 1, 1,
0, 4, 0]'),
 Text(2092.5, 217.44000000000005, 'gini = 0.765\nsamples = 6\nvalue = [0, 0,
0, 3, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0\n0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 2, 0]'),
 Text(2557.5, 217.44000000000005, 'gini = 0.72\nsamples = 5\nvalue = [0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0\n1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 2, 0]')]
```



Conclusion : RandomForestClassifier() 0.3857142857142857 HIGH RANGE

In [ ]:

In [ ]:

In [ ]: