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

In [2]:
```
data=pd.read_csv(r"C:\Users\user\Desktop\DINESH\C10_air\madrid_2007.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2007-12-01 01:00:00	NaN	2.86	NaN	NaN	NaN	282.200012	1054.000000	NaN	4.030000	156.1
1	2007-12-01 01:00:00	NaN	1.82	NaN	NaN	NaN	86.419998	354.600006	NaN	3.260000	80.8
2	2007-12-01 01:00:00	NaN	1.47	NaN	NaN	NaN	94.639999	319.000000	NaN	5.310000	53.0
3	2007-12-01 01:00:00	NaN	1.64	NaN	NaN	NaN	127.900002	476.700012	NaN	4.500000	105.3
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	106.5
...	...	...	...	...	...	...	...	...	...	...	
225115	2007-03-01 00:00:00	0.30	0.45	1.00	0.30	0.26	8.690000	11.690000	1.00	42.209999	6.7
225116	2007-03-01 00:00:00	NaN	0.16	NaN	NaN	NaN	46.820000	51.480000	NaN	22.150000	5.7
225117	2007-03-01 00:00:00	0.24	NaN	0.20	NaN	0.09	51.259998	66.809998	NaN	18.540001	13.0
225118	2007-03-01 00:00:00	0.11	NaN	1.00	NaN	0.05	24.240000	36.930000	NaN	NaN	6.6
225119	2007-03-01 00:00:00	0.53	0.40	1.00	1.70	0.12	32.360001	47.860001	1.37	24.150000	10.2

225120 rows × 17 columns

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

Out[3]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	2007-12-01 01:00:00	NaN	2.86	NaN	NaN	NaN	282.200012	1054.000000	NaN	4.030000	156.199997
1	2007-12-01 01:00:00	NaN	1.82	NaN	NaN	NaN	86.419998	354.600006	NaN	3.260000	80.809998
2	2007-12-01 01:00:00	NaN	1.47	NaN	NaN	NaN	94.639999	319.000000	NaN	5.310000	53.099998
3	2007-12-01 01:00:00	NaN	1.64	NaN	NaN	NaN	127.900002	476.700012	NaN	4.500000	105.300003
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	106.500000
5	2007-12-01 01:00:00	NaN	1.35	NaN	NaN	0.56	115.300003	319.600006	NaN	9.880000	57.500000
6	2007-12-01 01:00:00	5.54	1.87	4.65	NaN	0.75	165.100006	520.000000	NaN	4.780000	75.989998
7	2007-12-01 01:00:00	NaN	1.57	NaN	NaN	NaN	97.830002	369.000000	NaN	4.870000	59.590000
8	2007-12-01 01:00:00	NaN	0.70	NaN	NaN	NaN	107.699997	188.500000	NaN	4.560000	43.340000
9	2007-12-01 01:00:00	NaN	1.48	NaN	NaN	0.69	152.500000	485.200012	NaN	8.230000	80.830002

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

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM1
225100	2007-03-01 00:00:00	0.30	0.42	0.54	NaN	0.20	46.040001	87.620003	NaN	14.060000	23.80999
225101	2007-03-01 00:00:00	NaN	0.64	NaN	NaN	NaN	38.150002	54.330002	NaN	24.340000	7.18000
225102	2007-03-01 00:00:00	NaN	0.52	NaN	NaN	NaN	35.400002	49.939999	NaN	28.690001	5.24000
225103	2007-03-01 00:00:00	NaN	0.63	NaN	NaN	0.19	21.650000	29.620001	NaN	27.610001	6.81000
225104	2007-03-01 00:00:00	NaN	0.50	NaN	NaN	NaN	42.320000	66.809998	NaN	15.100000	11.82000
225105	2007-03-01 00:00:00	NaN	0.28	NaN	NaN	NaN	33.810001	46.759998	NaN	17.469999	9.70000
225106	2007-03-01 00:00:00	NaN	0.25	NaN	NaN	NaN	35.500000	38.680000	NaN	29.420000	12.66000
225107	2007-03-01 00:00:00	1.29	0.39	2.12	NaN	0.09	37.250000	87.760002	NaN	NaN	21.79999
225108	2007-03-01 00:00:00	NaN	0.44	NaN	NaN	NaN	21.570000	26.620001	NaN	29.580000	5.39000
225109	2007-03-01 00:00:00	NaN	0.26	NaN	NaN	NaN	15.960000	17.719999	NaN	25.150000	5.38000
225110	2007-03-01 00:00:00	NaN	0.28	NaN	NaN	NaN	23.920000	30.080000	NaN	26.469999	5.35000
225111	2007-03-01 00:00:00	NaN	0.33	NaN	NaN	NaN	29.230000	31.320000	NaN	19.040001	8.44000
225112	2007-03-01 00:00:00	NaN	0.58	NaN	NaN	NaN	30.280001	45.090000	NaN	26.320000	8.39000
225113	2007-03-01 00:00:00	NaN	0.23	NaN	NaN	NaN	10.840000	17.850000	NaN	33.970001	8.90000
225114	2007-03-01 00:00:00	0.54	0.28	0.59	NaN	0.01	31.440001	34.480000	NaN	26.049999	9.07000
225115	2007-03-01 00:00:00	0.30	0.45	1.00	0.3	0.26	8.690000	11.690000	1.00	42.209999	6.76000
225116	2007-03-01 00:00:00	NaN	0.16	NaN	NaN	NaN	46.820000	51.480000	NaN	22.150000	5.70000

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM1
225117	2007-03-01 00:00:00	0.24	NaN	0.20	NaN	0.09	51.259998	66.809998	NaN	18.540001	13.01000
225118	2007-03-01 00:00:00	0.11	NaN	1.00	NaN	0.05	24.240000	36.930000	NaN	NaN	6.61000
225119	2007-03-01 00:00:00	0.53	0.40	1.00	1.7	0.12	32.360001	47.860001	1.37	24.150000	10.26000

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

Out[5]:

	BEN	CO	EBE	MXY	NMHC	NO_2	
count	68885.000000	206748.000000	68883.000000	26061.000000	86883.000000	223985.000000	2
mean	0.925110	0.497212	1.374792	2.380600	0.226589	60.024280	
std	1.267360	0.391606	1.592087	2.791648	0.150634	38.003281	
min	0.100000	0.000000	0.100000	0.150000	0.000000	1.050000	
25%	0.200000	0.270000	0.690000	0.960000	0.130000	32.439999	
50%	0.490000	0.400000	1.000000	1.490000	0.210000	53.689999	
75%	1.150000	0.610000	1.420000	2.840000	0.290000	79.639999	
max	30.139999	9.660000	84.279999	65.480003	4.520000	628.599976	

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

Out[6]:
```
(225120, 17)
```

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

Out[7]:
```
3827040
```

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
...	...	...	...	...	...	...	...	...	...	...	...	...	...
225115	False	False	False	False	False	False	False	False	False	False	False	False	False
225116	False	True	False	True	True	True	False	False	True	False	False	True	True
225117	False	False	True	False	True	False	False	False	True	False	False	False	True
225118	False	False	True	False	True	False	False	False	True	True	False	True	True
225119	False	False	False	False	False	False	False	False	False	False	False	False	False

225120 rows × 17 columns

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

Out[9]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	P
4	2007-12-01 01:00:00	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024	3.49	52.689999	106.50
21	2007-12-01 01:00:00	1.98	0.31	2.56	6.06	0.35	76.059998	208.899994	1.70	1.000000	37.79
25	2007-12-01 01:00:00	2.82	1.42	3.15	7.02	0.49	123.099998	402.399994	2.60	7.160000	70.80
30	2007-12-01 02:00:00	4.65	1.89	4.41	8.21	0.65	151.000000	622.700012	3.55	58.080002	117.09
47	2007-12-01 02:00:00	1.97	0.30	2.15	5.08	0.33	78.760002	189.800003	1.62	1.000000	34.74
...	...	...	...	...	...	...	...	...	...	...	
225073	2007-02-28 23:00:00	2.12	0.47	2.51	4.99	0.05	43.560001	83.889999	2.57	13.090000	21.86
225094	2007-02-28 23:00:00	0.87	0.45	1.19	2.66	0.13	40.000000	61.959999	1.79	20.440001	15.07
225098	2007-03-01 00:00:00	0.95	0.41	1.55	3.11	0.05	36.090000	63.349998	1.74	17.160000	9.21
225115	2007-03-01 00:00:00	0.30	0.45	1.00	0.30	0.26	8.690000	11.690000	1.00	42.209999	6.76
225119	2007-03-01 00:00:00	0.53	0.40	1.00	1.70	0.12	32.360001	47.860001	1.37	24.150000	10.26

25443 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']]
```

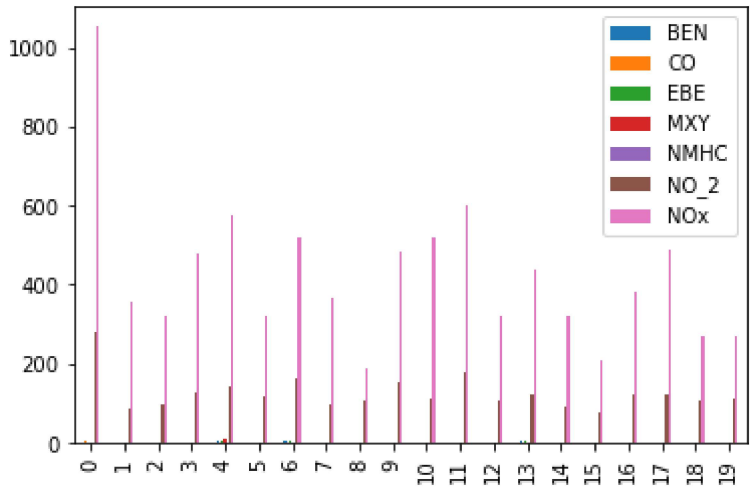
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx
0	NaN	2.86	NaN	NaN	NaN	282.200012	1054.000000
1	NaN	1.82	NaN	NaN	NaN	86.419998	354.600006
2	NaN	1.47	NaN	NaN	NaN	94.639999	319.000000
3	NaN	1.64	NaN	NaN	NaN	127.900002	476.700012
4	4.64	1.86	4.26	7.98	0.57	145.100006	573.900024
5	NaN	1.35	NaN	NaN	0.56	115.300003	319.600006
6	5.54	1.87	4.65	NaN	0.75	165.100006	520.000000
7	NaN	1.57	NaN	NaN	NaN	97.830002	369.000000
8	NaN	0.70	NaN	NaN	NaN	107.699997	188.500000
9	NaN	1.48	NaN	NaN	0.69	152.500000	485.200012
10	NaN	1.87	NaN	NaN	NaN	113.500000	519.099976
11	NaN	1.56	NaN	NaN	NaN	178.500000	599.099976
12	NaN	1.21	NaN	NaN	NaN	104.599998	322.299988
13	4.96	1.51	7.52	NaN	0.72	124.900002	436.500000
14	NaN	0.99	NaN	NaN	NaN	89.930000	323.100006
15	NaN	1.09	NaN	NaN	NaN	74.739998	207.800003
16	NaN	1.13	NaN	NaN	NaN	120.000000	382.200012
17	NaN	2.17	NaN	NaN	NaN	120.699997	489.600006
18	NaN	0.93	NaN	NaN	NaN	104.699997	268.600006
19	NaN	1.02	NaN	NaN	NaN	110.800003	268.799988

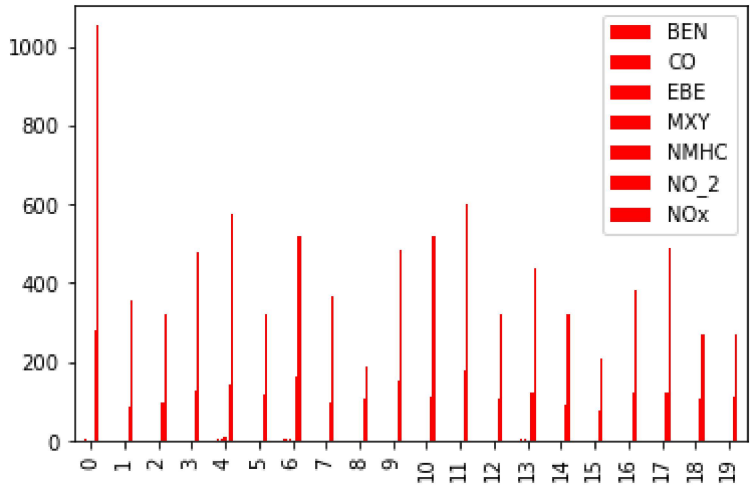
In [13]:
```
dd.plot.bar()
```

Out[13]: <AxesSubplot:>



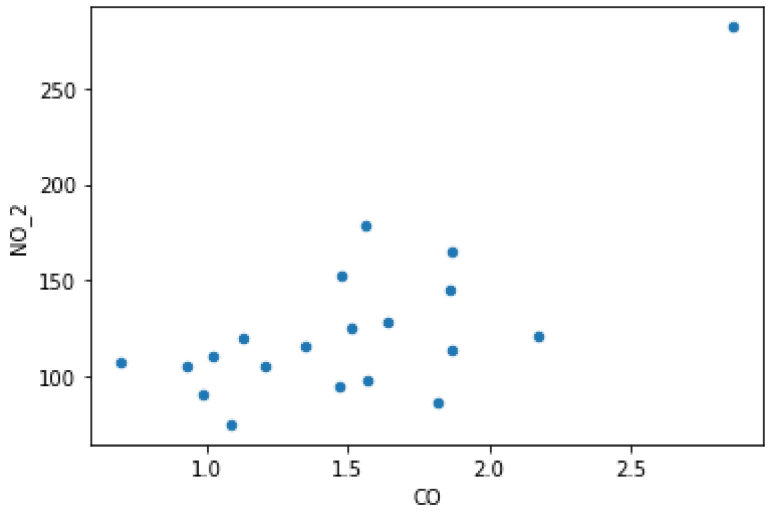
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



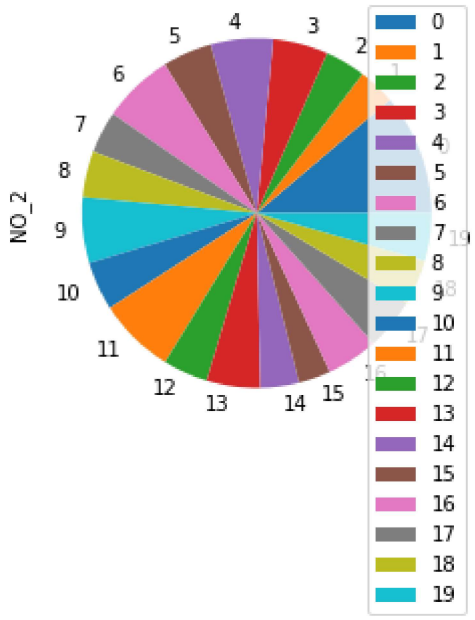
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

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



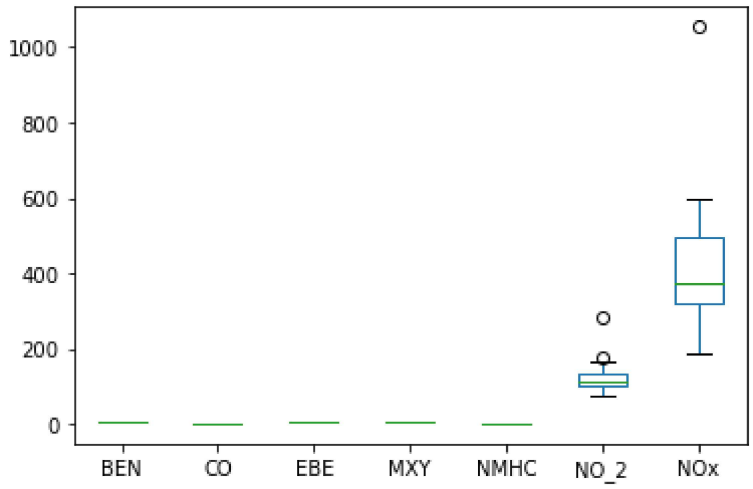
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



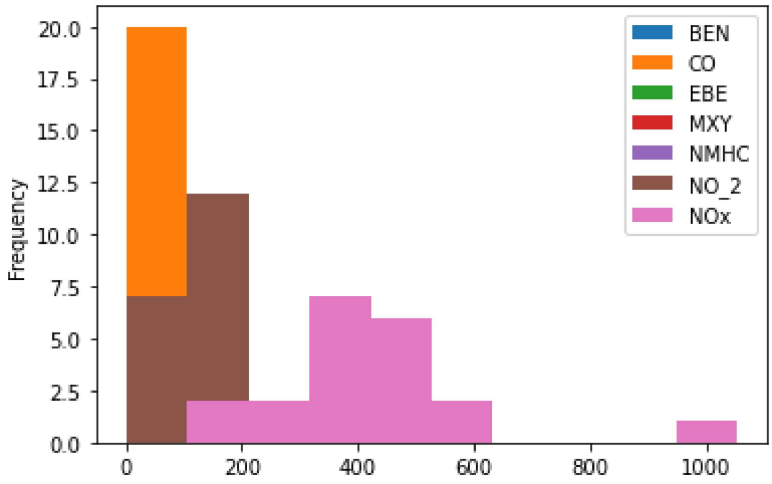
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



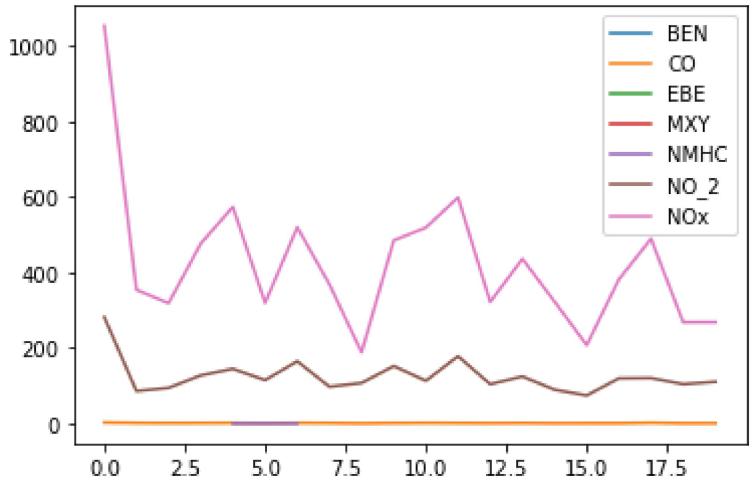
In [18]:
```
dd.plot.hist()
```

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



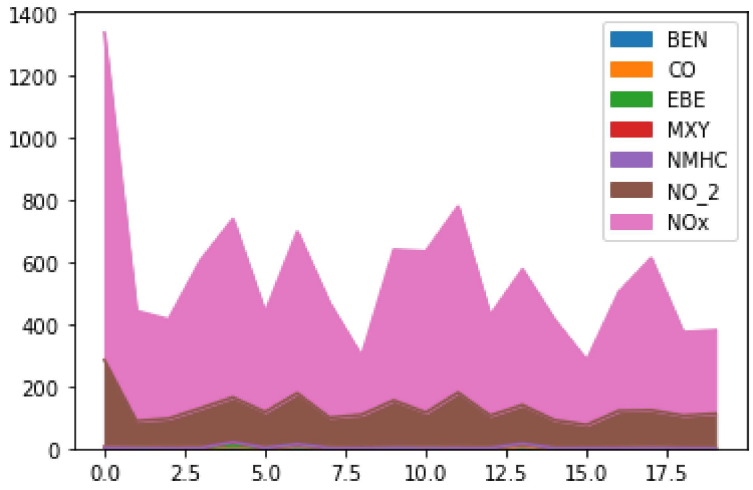
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



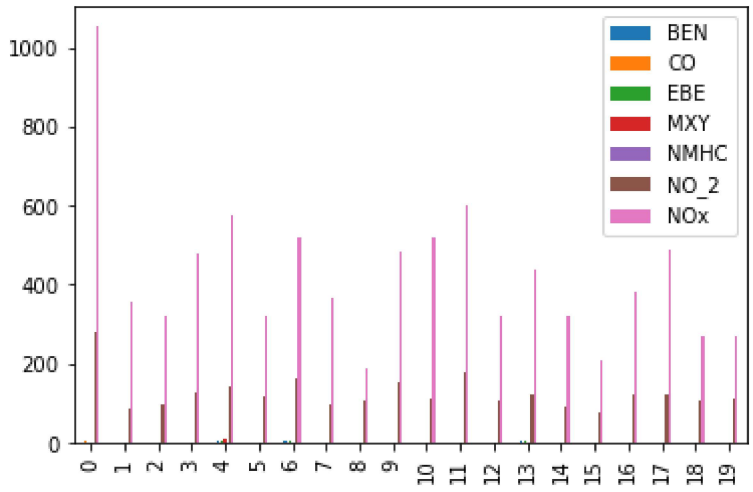
In [20]:
```
dd.plot.area()
```

Out[20]: <AxesSubplot:>



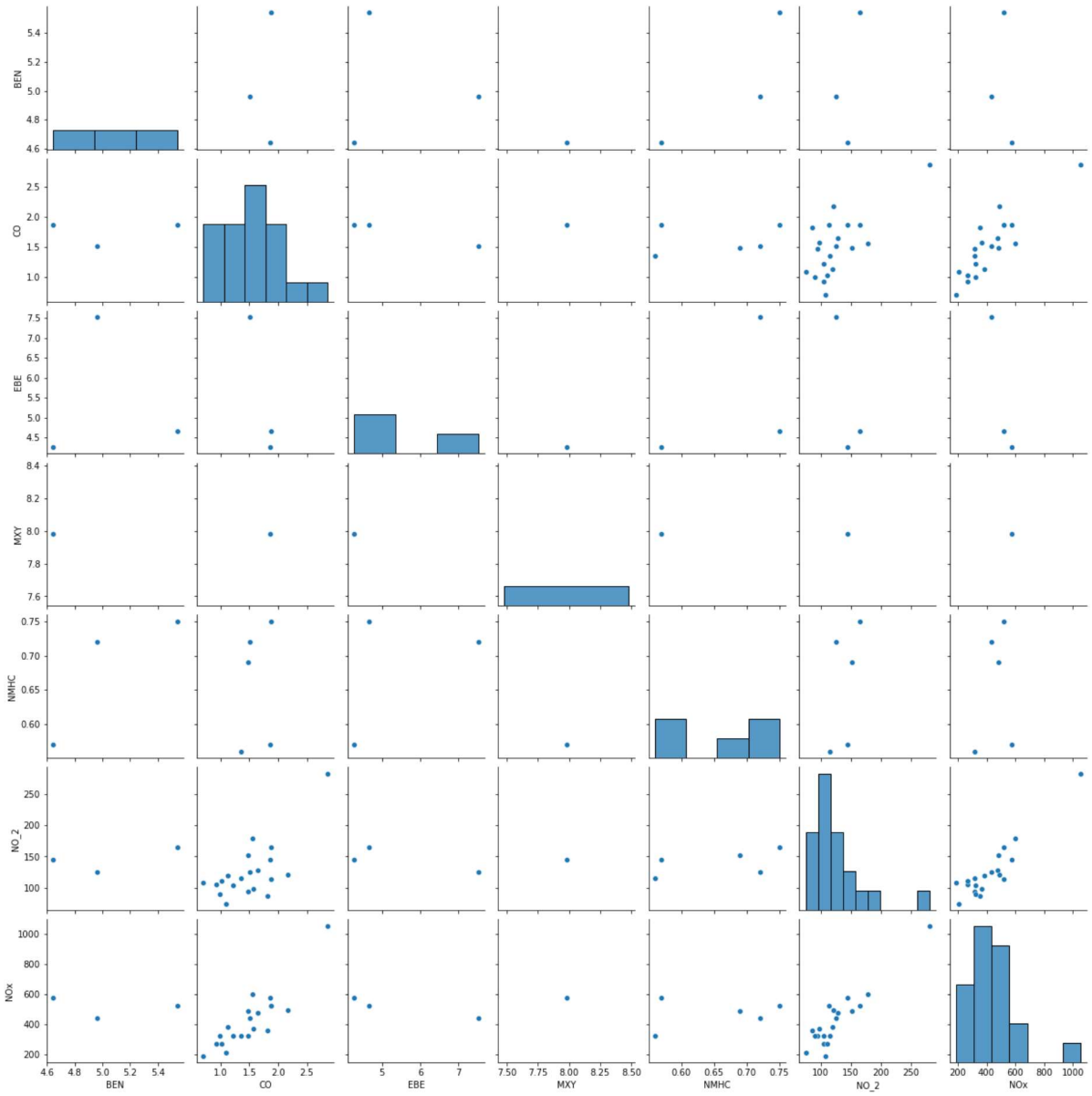
In [21]:
```
dd.plot.bar()
```

Out[21]: <AxesSubplot:>



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

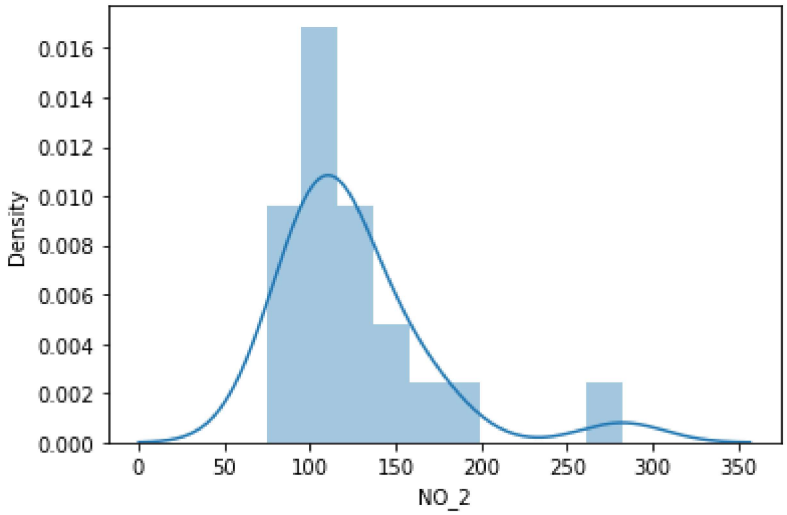
Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x1ff823f1850>
```



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

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

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



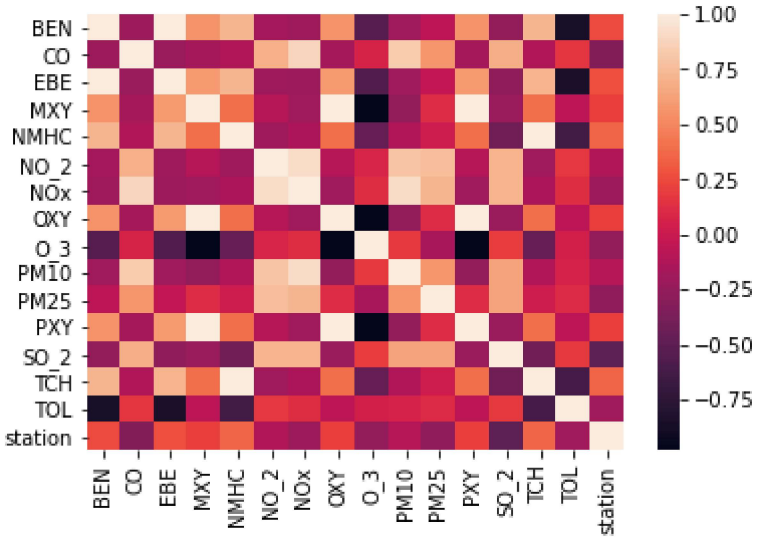
In [24]:
```
ds=data.fillna(20)
```

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_)
```

```
28078941.533974882
```

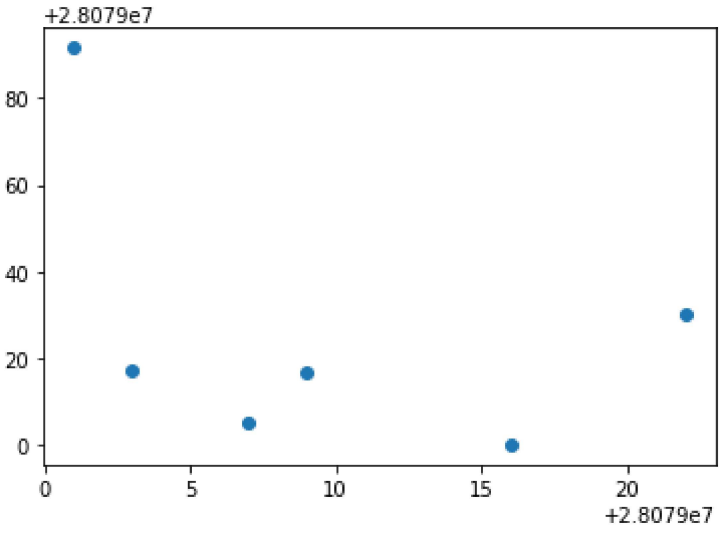
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	-10.575983
CO	31.584053
EBE	11.268825
MXY	-1.804603
NMHC	1.532780
NO_2	0.896020
NOx	-0.191145

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

Out[33]:
```
<matplotlib.collections.PathCollection at 0x1ff8989e2e0>
```



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

```
-26.566290325403603
```

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

Out[35]:
```
-26.566290325403603
```

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

Out[36]:
```
0.7233092971807642
```

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]:
```
-12.837119538450569
```

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

Out[40]:
```
0.553610865347545
```

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]:
```
-9.02430573960763
```

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

Out[43]:
```
0.4930110106646767
```

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_)
```

```
[-1.04481682  0.          1.02689292  0.          1.15591615  0.38300964
 -0.03443677]
```

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

```
28078970.3233885
```

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

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

```
-12.487721357414491
```

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)
```

```
[28079001]
```

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]:
```
0.8791802763600968
```

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')
```

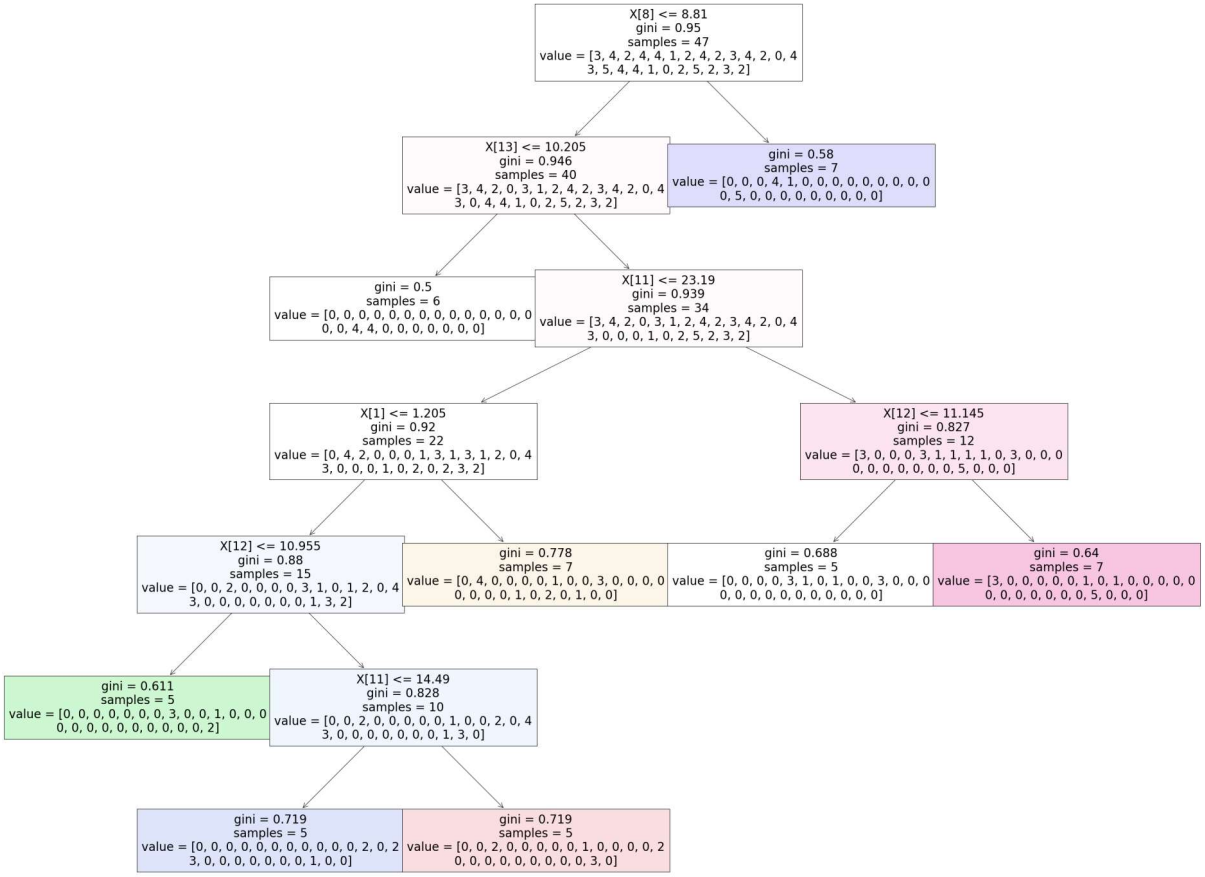
In [70]:
```
grid_search.best_score_
```

Out[70]:
```
0.41428571428571426
```

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

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

Out[72]:
```
[Text(1550.0, 2019.0857142857144, 'X[8] <= 8.81\ngini = 0.95\nsamples = 47\nv
alue = [3, 4, 2, 4, 4, 1, 2, 4, 2, 3, 4, 2, 0, 4\n3, 5, 4, 4, 1, 0, 2, 5, 2,
3, 2]'),
 Text(1240.0, 1708.457142857143, 'X[13] <= 10.205\ngini = 0.946\nsamples = 40
\nvalue = [3, 4, 2, 0, 3, 1, 2, 4, 2, 3, 4, 2, 0, 4\n3, 0, 4, 4, 1, 0, 2, 5,
2, 3, 2]'),
 Text(930.0, 1397.8285714285716, 'gini = 0.5\nsamples = 6\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 4, 4, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1550.0, 1397.8285714285716, 'X[11] <= 23.19\ngini = 0.939\nsamples = 34
\nvalue = [3, 4, 2, 0, 3, 1, 2, 4, 2, 3, 4, 2, 0, 4\n3, 0, 0, 0, 1, 0, 2, 5,
2, 3, 2]'),
 Text(930.0, 1087.2, 'X[1] <= 1.205\ngini = 0.92\nsamples = 22\nvalue = [0,
4, 2, 0, 0, 0, 1, 3, 1, 3, 1, 2, 0, 4\n3, 0, 0, 0, 1, 0, 2, 0, 2, 3, 2]'),
 Text(620.0, 776.5714285714287, 'X[12] <= 10.955\ngini = 0.88\nsamples = 15\n
value = [0, 0, 2, 0, 0, 0, 0, 3, 1, 0, 1, 2, 0, 4\n3, 0, 0, 0, 0, 0, 0, 0, 1,
3, 2]'),
 Text(310.0, 465.9428571428573, 'gini = 0.611\nsamples = 5\nvalue = [0, 0, 0,
0, 0, 0, 0, 3, 0, 0, 1, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2]'),
 Text(930.0, 465.9428571428573, 'X[11] <= 14.49\ngini = 0.828\nsamples = 10\n
value = [0, 0, 2, 0, 0, 0, 0, 0, 1, 0, 0, 2, 0, 4\n3, 0, 0, 0, 0, 0, 0, 0, 1,
3, 0]'),
 Text(620.0, 155.3142857142857, 'gini = 0.719\nsamples = 5\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2\n3, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]'),
 Text(1240.0, 155.3142857142857, 'gini = 0.719\nsamples = 5\nvalue = [0, 0,
2, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 2\n0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0]'),
 Text(1240.0, 776.5714285714287, 'gini = 0.778\nsamples = 7\nvalue = [0, 4,
0, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0, 0\n0, 0, 0, 0, 1, 0, 2, 0, 1, 0, 0]'),
 Text(2170.0, 1087.2, 'X[12] <= 11.145\ngini = 0.827\nsamples = 12\nvalue =
[3, 0, 0, 0, 3, 1, 1, 1, 1, 0, 3, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 5, 0, 0,
0]'),
 Text(1860.0, 776.5714285714287, 'gini = 0.688\nsamples = 5\nvalue = [0, 0,
0, 0, 3, 1, 0, 1, 0, 0, 3, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(2480.0, 776.5714285714287, 'gini = 0.64\nsamples = 7\nvalue = [3, 0, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0]'),
 Text(1860.0, 1708.457142857143, 'gini = 0.58\nsamples = 7\nvalue = [0, 0, 0,
4, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0]')]
```



Conclusion : LogisticRegression() [28079001] HIGH RANGE

In [ ]: