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_2001.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
0	2001-08-01 01:00:00	NaN	0.37	NaN	NaN	NaN	58.400002	87.150002	NaN	34.529999	105.0
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	100.5
2	2001-08-01 01:00:00	NaN	0.28	NaN	NaN	NaN	50.660000	61.380001	NaN	46.310001	100.0
3	2001-08-01 01:00:00	NaN	0.47	NaN	NaN	NaN	69.790001	73.449997	NaN	40.650002	69.7
4	2001-08-01 01:00:00	NaN	0.39	NaN	NaN	NaN	22.830000	24.799999	NaN	66.309998	75.1
...	...	...	...	...	...	...	...	...	...	...	
217867	2001-04-01 00:00:00	10.45	1.81	NaN	NaN	NaN	73.000000	264.399994	NaN	5.200000	47.8
217868	2001-04-01 00:00:00	5.20	0.69	4.56	NaN	0.13	71.080002	129.300003	NaN	13.460000	26.8
217869	2001-04-01 00:00:00	0.49	1.09	NaN	1.00	0.19	76.279999	128.399994	0.35	5.020000	40.7
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.38	NaN	80.019997	197.000000	2.58	5.840000	37.8
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.04	0.18	76.809998	206.300003	5.20	8.340000	35.3

217872 rows × 16 columns

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

Out[3]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
0	2001-08-01 01:00:00	NaN	0.37	NaN	NaN	NaN	58.400002	87.150002	NaN	34.529999	105.000000
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	100.599998
2	2001-08-01 01:00:00	NaN	0.28	NaN	NaN	NaN	50.660000	61.380001	NaN	46.310001	100.099998
3	2001-08-01 01:00:00	NaN	0.47	NaN	NaN	NaN	69.790001	73.449997	NaN	40.650002	69.779999
4	2001-08-01 01:00:00	NaN	0.39	NaN	NaN	NaN	22.830000	24.799999	NaN	66.309998	75.180000
5	2001-08-01 01:00:00	2.11	0.63	2.48	5.94	0.05	66.260002	118.099998	3.15	33.500000	122.699997
6	2001-08-01 01:00:00	NaN	0.28	NaN	NaN	NaN	35.799999	39.590000	NaN	68.250000	124.900002
7	2001-08-01 01:00:00	NaN	0.67	NaN	NaN	NaN	74.830002	112.000000	NaN	26.410000	113.000000
8	2001-08-01 01:00:00	NaN	0.41	NaN	NaN	NaN	33.209999	37.299999	NaN	62.299999	125.300003
9	2001-08-01 01:00:00	NaN	0.17	NaN	NaN	0.13	24.129999	36.970001	NaN	46.200001	95.589996

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

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
217852	2001-04-01 00:00:00	NaN	0.95	NaN	NaN	NaN	47.720001	79.470001	NaN	7.91	
217853	2001-04-01 00:00:00	14.47	1.83	11.39	26.059999	0.33	84.230003	259.200012	11.39	5.44	36
217854	2001-04-01 00:00:00	12.64	1.45	7.27	NaN	NaN	91.750000	167.600006	NaN	6.88	22
217855	2001-04-01 00:00:00	NaN	3.80	NaN	NaN	NaN	120.900002	471.899994	NaN	2.07	66
217856	2001-04-01 00:00:00	NaN	2.64	NaN	NaN	NaN	93.820000	210.199997	NaN	8.33	20
217857	2001-04-01 00:00:00	NaN	0.62	NaN	NaN	NaN	61.380001	85.029999	NaN	14.65	8
217858	2001-04-01 00:00:00	NaN	0.77	NaN	NaN	0.01	58.939999	62.330002	NaN	12.48	7
217859	2001-04-01 00:00:00	NaN	1.14	NaN	NaN	NaN	90.180000	168.300003	NaN	4.51	30
217860	2001-04-01 00:00:00	NaN	2.60	NaN	NaN	NaN	89.559998	246.000000	NaN	12.14	50
217861	2001-04-01 00:00:00	9.97	0.95	6.04	NaN	0.22	79.129997	177.800003	NaN	7.65	25
217862	2001-04-01 00:00:00	NaN	0.78	NaN	NaN	NaN	41.950001	58.310001	NaN	12.70	
217863	2001-04-01 00:00:00	NaN	3.39	NaN	NaN	NaN	19.480000	364.799988	NaN	13.57	103
217864	2001-04-01 00:00:00	NaN	1.68	NaN	NaN	NaN	79.940002	287.799988	NaN	5.36	44
217865	2001-04-01 00:00:00	NaN	1.24	NaN	NaN	NaN	100.300003	169.300003	NaN	9.19	20
217866	2001-04-01 00:00:00	NaN	1.10	NaN	NaN	NaN	75.000000	164.600006	NaN	13.82	34
217867	2001-04-01 00:00:00	10.45	1.81	NaN	NaN	NaN	73.000000	264.399994	NaN	5.20	47
217868	2001-04-01 00:00:00	5.20	0.69	4.56	NaN	0.13	71.080002	129.300003	NaN	13.46	26

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	
217869	2001-04-01 00:00:00	0.49	1.09	NaN	1.000000	0.19	76.279999	128.399994	0.35	5.02	40
217870	2001-04-01 00:00:00	5.62	1.01	5.04	11.380000	NaN	80.019997	197.000000	2.58	5.84	37
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.040000	0.18	76.809998	206.300003	5.20	8.34	35

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

Out[5]:

	BEN	CO	EBE	MXY	NMHC	NO_2	
count	70389.000000	216341.000000	57752.000000	42753.000000	85719.00000	216331.000000	21
mean	3.276697	0.921095	3.318622	6.882131	0.19747	62.938905	
std	3.755713	0.928380	3.722382	7.761617	0.20974	36.008389	
min	0.100000	0.000000	0.140000	0.190000	0.00000	0.010000	
25%	0.960000	0.380000	1.170000	2.000000	0.08000	36.650002	
50%	2.100000	0.650000	2.290000	4.560000	0.14000	58.709999	
75%	4.210000	1.130000	4.150000	8.960000	0.25000	82.790001	
max	63.000000	18.040001	126.500000	163.699997	4.98000	586.099976	

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

Out[6]:
```
(217872, 16)
```

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

Out[7]:
```
3485952
```

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

Out[8]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10	PXY	SO_2
0	False	True	False	True	True	True	False	False	True	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	True	False	True	True	True	False	False	True	False	False	True	False
3	False	True	False	True	True	True	False	False	True	False	False	True	False
4	False	True	False	True	True	True	False	False	True	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
217867	False	False	False	True	True	True	False	False	True	False	False	True	False
217868	False	False	False	False	True	False	False	False	True	False	False	True	False
217869	False	False	False	True	False	False	False	False	False	False	False	False	False
217870	False	False	False	False	False	True	False	False	False	False	False	False	False
217871	False	False	False	False	False	False	False	False	False	False	False	False	False

217872 rows × 16 columns

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

Out[9]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.100000	0.07	56.250000	75.169998	2.11	42.160000
5	2001-08-01 01:00:00	2.11	0.63	2.48	5.940000	0.05	66.260002	118.099998	3.15	33.500000
21	2001-08-01 01:00:00	0.80	0.43	0.71	1.200000	0.10	27.190001	29.700001	0.76	56.990002
23	2001-08-01 01:00:00	1.29	0.34	1.41	3.090000	0.07	40.750000	51.570000	1.70	51.580002
25	2001-08-01 02:00:00	0.87	0.06	0.88	2.410000	0.01	29.709999	31.440001	1.20	56.520000
...	...	...	...	...	...	...	...	...	...	...
217829	2001-03-31 23:00:00	11.76	4.48	7.71	17.219999	0.89	103.900002	548.500000	7.62	9.680000
217847	2001-03-31 23:00:00	9.79	2.65	7.59	9.730000	0.46	91.320000	315.899994	3.75	6.660000
217849	2001-04-01 00:00:00	5.86	1.22	5.66	13.710000	0.25	64.370003	218.300003	6.46	7.480000
217853	2001-04-01 00:00:00	14.47	1.83	11.39	26.059999	0.33	84.230003	259.200012	11.39	5.440000
217871	2001-04-01 00:00:00	8.09	1.62	6.66	13.040000	0.18	76.809998	206.300003	5.20	8.340000

29669 rows × 16 columns

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

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

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

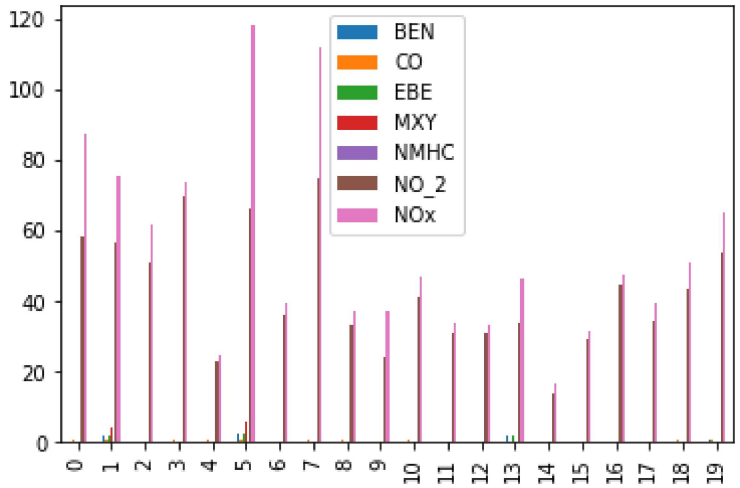
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx
0	NaN	0.37	NaN	NaN	NaN	58.400002	87.150002
1	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998
2	NaN	0.28	NaN	NaN	NaN	50.660000	61.380001
3	NaN	0.47	NaN	NaN	NaN	69.790001	73.449997
4	NaN	0.39	NaN	NaN	NaN	22.830000	24.799999
5	2.11	0.63	2.48	5.94	0.05	66.260002	118.099998
6	NaN	0.28	NaN	NaN	NaN	35.799999	39.590000
7	NaN	0.67	NaN	NaN	NaN	74.830002	112.000000
8	NaN	0.41	NaN	NaN	NaN	33.209999	37.299999
9	NaN	0.17	NaN	NaN	0.13	24.129999	36.970001
10	NaN	0.38	NaN	NaN	0.02	40.900002	46.610001
11	NaN	0.17	NaN	NaN	0.09	30.629999	33.770000
12	NaN	0.18	NaN	NaN	NaN	30.920000	33.020000
13	1.63	0.24	1.81	NaN	0.10	33.869999	45.970001
14	NaN	0.25	NaN	NaN	NaN	13.970000	16.500000
15	NaN	0.17	NaN	NaN	0.06	28.980000	31.480000
16	NaN	0.26	NaN	NaN	NaN	44.770000	47.580002
17	NaN	0.29	NaN	NaN	NaN	34.070000	39.419998
18	NaN	0.33	NaN	NaN	NaN	43.209999	50.939999
19	0.83	0.81	NaN	NaN	NaN	53.480000	65.089996

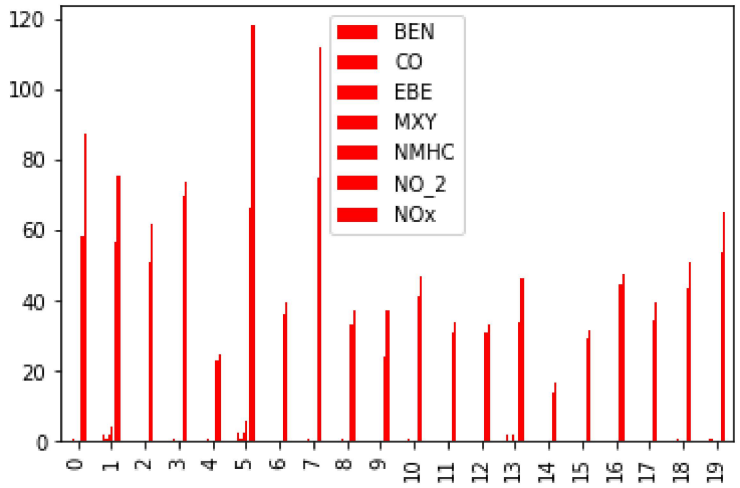
In [13]:
```
dd.plot.bar()
```

Out[13]: <AxesSubplot:>



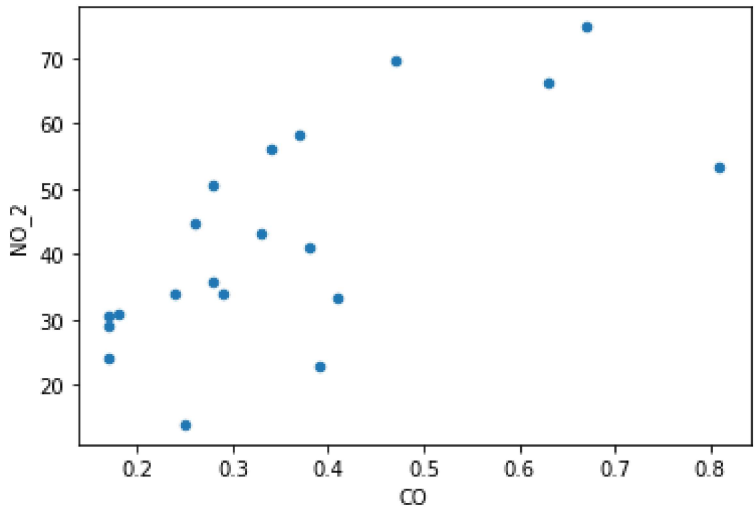
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



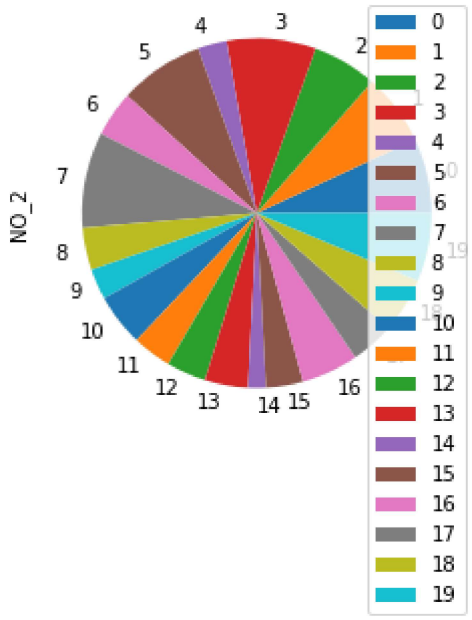
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

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



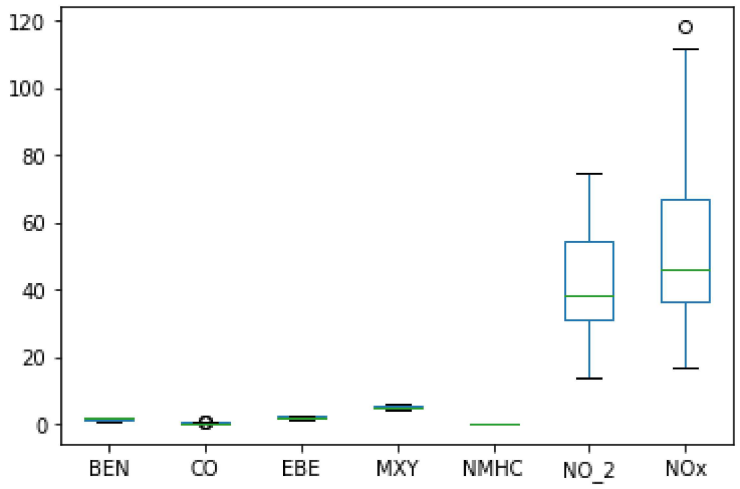
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



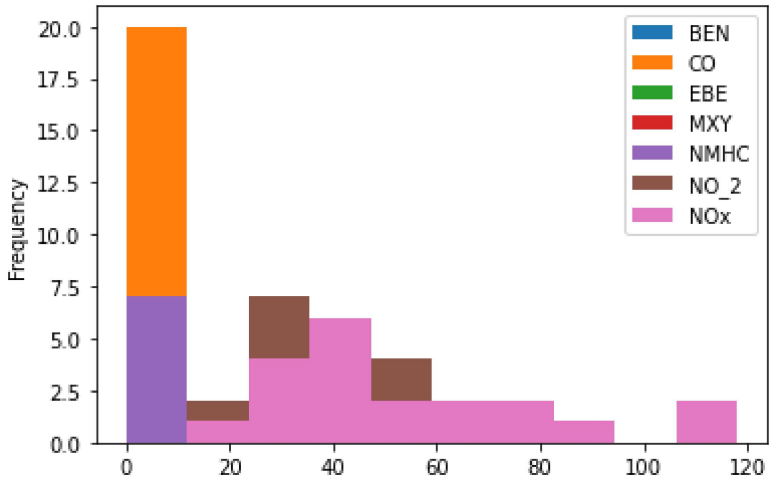
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



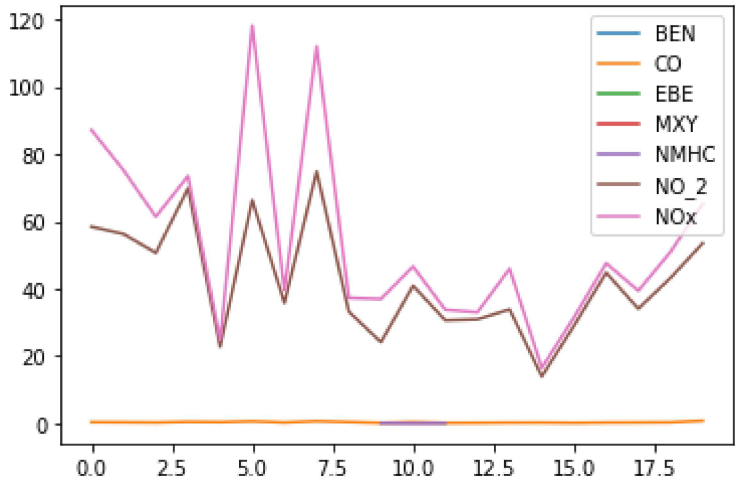
In [18]:
```
dd.plot.hist()
```

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



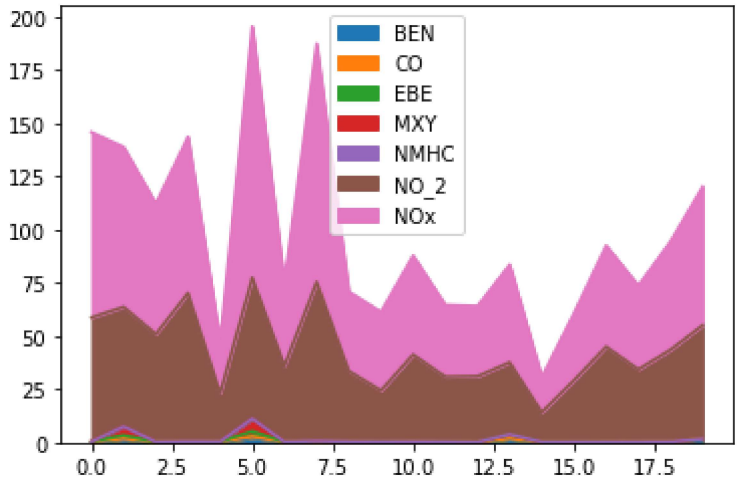
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



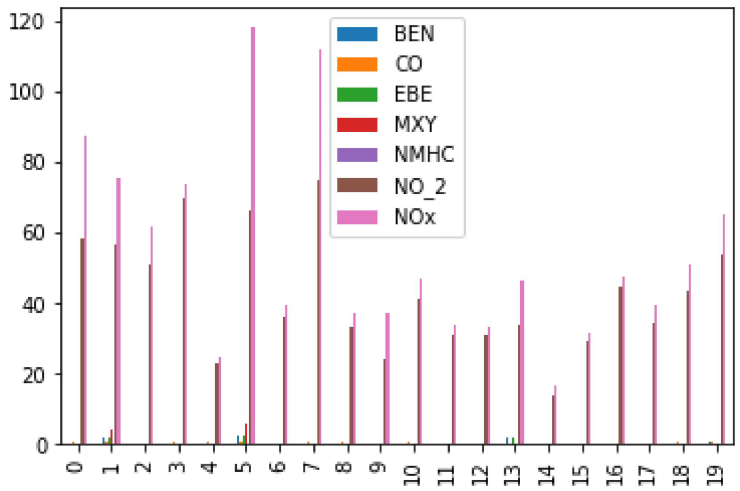
In [20]:
```
dd.plot.area()
```

Out[20]: <AxesSubplot:>



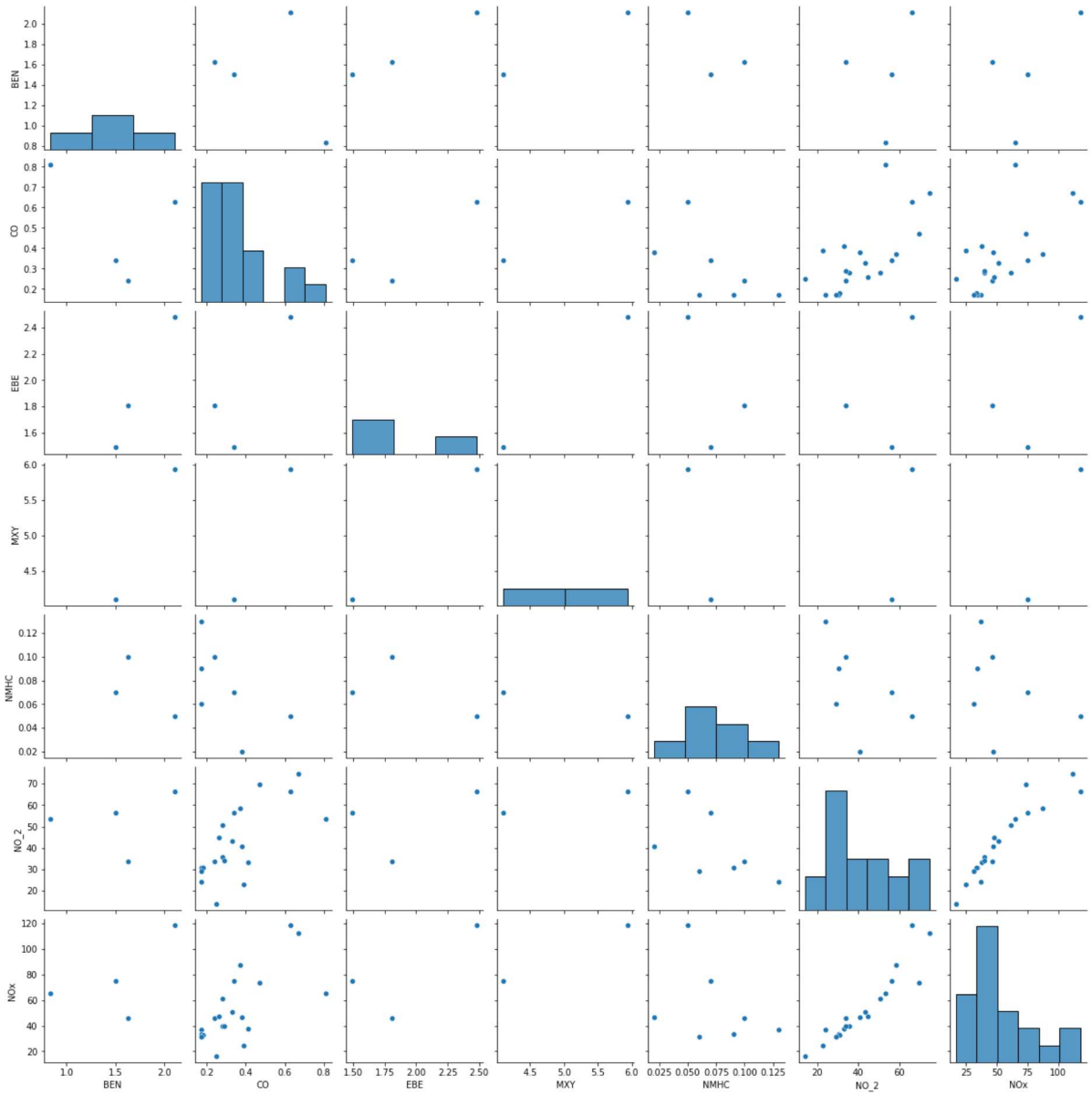
In [21]:
```
dd.plot.bar()
```

Out[21]: <AxesSubplot:>



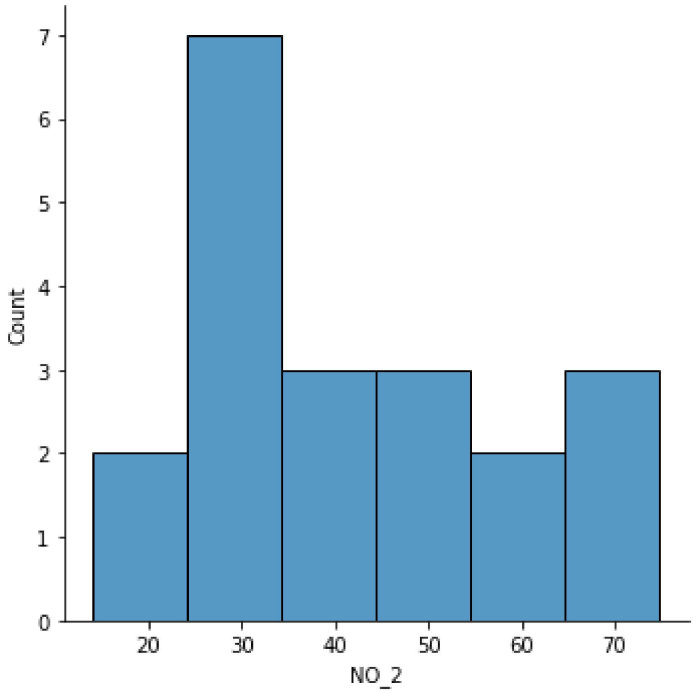
In [22]:
```
sns.pairplot(dd)
```

Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x1d873470280>
```



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

Out[23]:
```
<seaborn.axisgrid.FacetGrid at 0x1d871e6a9d0>
```



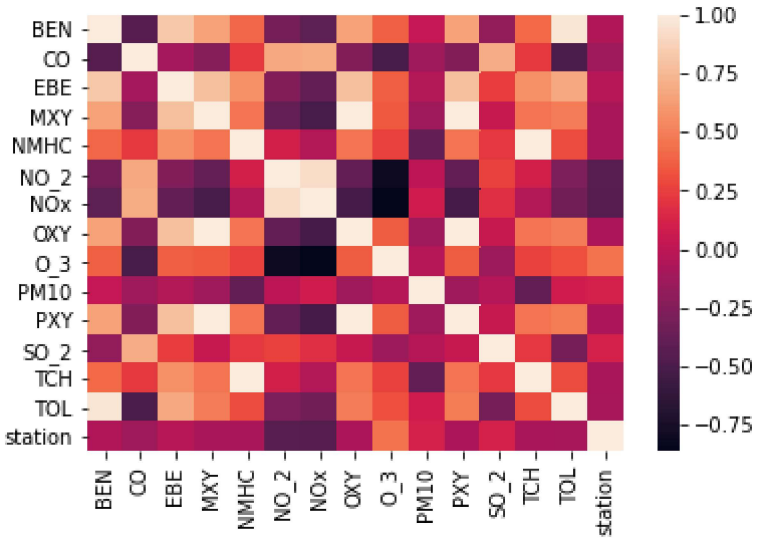
In [24]:
```
ds=data.fillna(20)
```

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)
```

LinearRegression

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

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

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

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

```
28079022.032701574
```

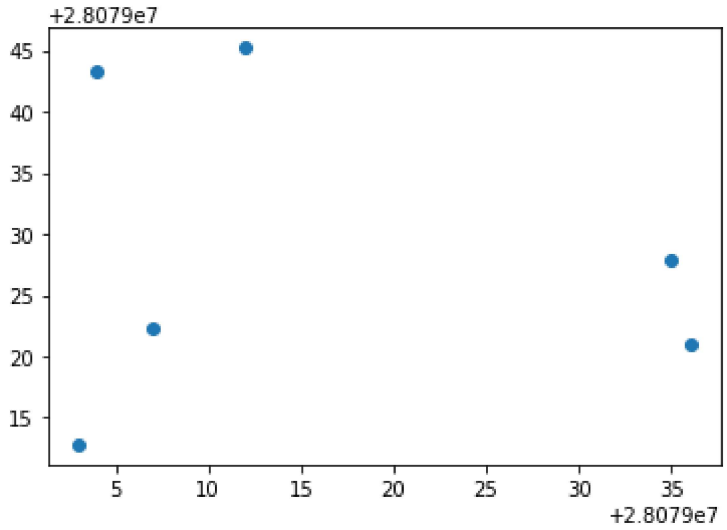
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	2.193674
CO	98.929082
EBE	-1.866319
MXY	-0.732622
NMHC	-0.779347
NO_2	1.553329
NOx	-1.497807

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

Out[33]: <matplotlib.collections.PathCollection at 0x1d87916a1f0>



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

```
-1.7779574530625837
```

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

Out[35]: -1.7779574530625837

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

Out[36]: 0.8063131442905054

Ridge,Lasso

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]: -0.42695082436791765

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

Out[40]: 0.4694871920000462

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]:
```
-0.05872908360293261
```

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

Out[43]:
```
0.33572669225059215
```

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_)
```

```
[-0.01070932  0.          0.14938975 -1.05602186 -0.04958988  0.91141722
 -0.77524011]
```

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

```
28079042.068914108
```

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

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

```
-0.3989799499573503
```

LogisticRegression()

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.iloc[:,0:20]
target_vector=ssd.iloc[:,-1]
```

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

Out[52]:
```
(20, 16)
```

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

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

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

In [55]:
```
feature_matrix = ssd[['BEN','CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx']]
target_vector=ssd['station']
```

In [56]:
```
feature_matrix.shape
```

Out[56]:
```
(20, 7)
```

In [57]:
```
target_vector.shape
```

Out[57]:
```
(20,)
```

In [58]:
```
from sklearn.preprocessing import StandardScaler
```

In [59]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [60]:
```
logr= LogisticRegression()
logr.fit(fs,target_vector)
```

Out[60]:
```
LogisticRegression()
```

In [61]:
```
observation =[[1.2,2.3,3.3,4.3,5.3,6.3,7.3]]
```

In [62]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
[28079009]
```

In [63]:
```
logr.classes_
```

Out[63]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079009,
       28079011, 28079012, 28079014, 28079015, 28079016, 28079018,
       28079019, 28079021, 28079022, 28079035, 28079036, 28079038,
       28079039, 28079040], dtype=int64)
```

In [64]:
```
logr.predict_proba(observation)[0][0]
```

Out[64]:
```
0.04816805334142426
```

In [65]:
```
ssd
```

Out[65]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	P
0	2001-08-01 01:00:00	20.00	0.37	20.00	20.00	20.00	58.400002	87.150002	20.00	34.529999	105.00
1	2001-08-01 01:00:00	1.50	0.34	1.49	4.10	0.07	56.250000	75.169998	2.11	42.160000	100.59
2	2001-08-01 01:00:00	20.00	0.28	20.00	20.00	20.00	50.660000	61.380001	20.00	46.310001	100.09
3	2001-08-01 01:00:00	20.00	0.47	20.00	20.00	20.00	69.790001	73.449997	20.00	40.650002	69.77
4	2001-08-01 01:00:00	20.00	0.39	20.00	20.00	20.00	22.830000	24.799999	20.00	66.309998	75.18

RandomForestClassifier()

In [66]:
```
ged=data[['BEN','CO','EBE','MXY','NMHC','NO_2','NOx','OXY','O_3','PM10','PXY',
```

In [67]:
```
d=ged.fillna(20)
```

In [68]:
```
dg=d.head(100)
```

In [69]:
```
x=dg[['BEN','CO','EBE','MXY','NMHC','NO_2','NOx','OXY','O_3','PM10','PXY','SO_2
y=dg['station']
```

In [70]:
```
print(len(x))
print(len(y))
```

```
100
100
```

In [71]:
```
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 [72]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[72]:
```
RandomForestClassifier()
```

In [73]:
```
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 [74]:
```
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[74]:
```
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')
```

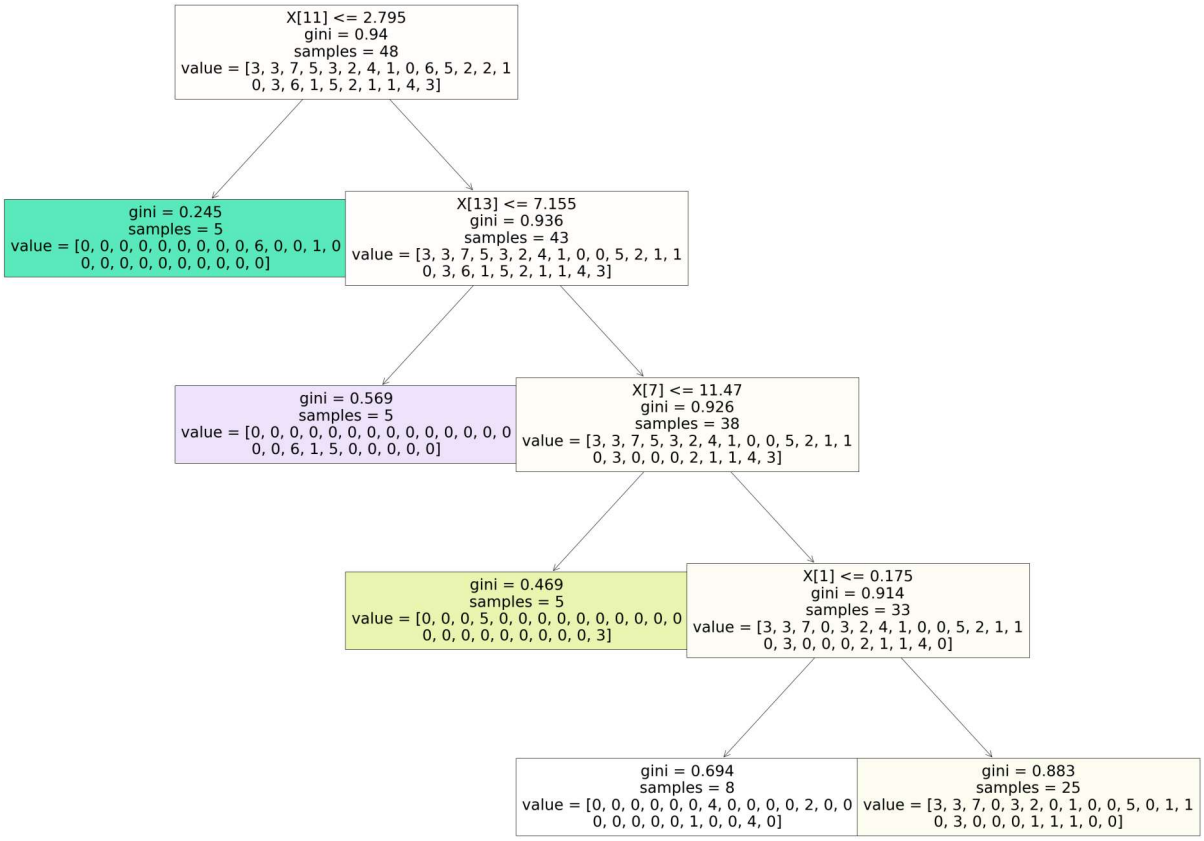
In [75]:
```
grid_search.best_score_
```

Out[75]:
```
0.4
```

In [76]:
```
rfc_best=grid_search.best_estimator_
```

In [77]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(50,40))
plot_tree(rfc_best.estimators_[5],filled=True)
```

Out[77]:
```
[Text(797.1428571428571, 1956.96, 'X[11] <= 2.795\ngini = 0.94\nsamples = 48
\nvalue = [3, 3, 7, 5, 3, 2, 4, 1, 0, 6, 5, 2, 2, 1\n0, 3, 6, 1, 5, 2, 1, 1,
4, 3]'),
 Text(398.57142857142856, 1522.0800000000002, 'gini = 0.245\nsamples = 5\nval
ue = [0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 1, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
0]'),
 Text(1195.7142857142858, 1522.0800000000002, 'X[13] <= 7.155\ngini = 0.936\n
samples = 43\nvalue = [3, 3, 7, 5, 3, 2, 4, 1, 0, 0, 5, 2, 1, 1\n0, 3, 6, 1,
5, 2, 1, 1, 4, 3]'),
 Text(797.1428571428571, 1087.2, 'gini = 0.569\nsamples = 5\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 6, 1, 5, 0, 0, 0, 0, 0]'),
 Text(1594.2857142857142, 1087.2, 'X[7] <= 11.47\ngini = 0.926\nsamples = 38
\nvalue = [3, 3, 7, 5, 3, 2, 4, 1, 0, 0, 5, 2, 1, 1\n0, 3, 0, 0, 0, 2, 1, 1,
4, 3]'),
 Text(1195.7142857142858, 652.3200000000002, 'gini = 0.469\nsamples = 5\nvalu
e = [0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0,
3]'),
 Text(1992.8571428571427, 652.3200000000002, 'X[1] <= 0.175\ngini = 0.914\nsa
mples = 33\nvalue = [3, 3, 7, 0, 3, 2, 4, 1, 0, 0, 5, 2, 1, 1\n0, 3, 0, 0, 0,
2, 1, 1, 4, 0]'),
 Text(1594.2857142857142, 217.44000000000005, 'gini = 0.694\nsamples = 8\nval
ue = [0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 2, 0, 0\n0, 0, 0, 0, 0, 1, 0, 0, 4,
0]'),
 Text(2391.4285714285716, 217.44000000000005, 'gini = 0.883\nsamples = 25\nva
lue = [3, 3, 7, 0, 3, 2, 0, 1, 0, 0, 5, 0, 1, 1\n0, 3, 0, 0, 0, 1, 1, 1, 0,
0]')]
```



Conclusion : LogisticRegression() [28079009] HIGH RANGE

In [ ]: