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_2003.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM
0	2003-03-01 01:00:00	NaN	1.72	NaN	NaN	NaN	73.900002	316.299988	NaN	10.550000	55.2099
1	2003-03-01 01:00:00	NaN	1.45	NaN	NaN	0.26	72.110001	250.000000	0.73	6.720000	52.3899
2	2003-03-01 01:00:00	NaN	1.57	NaN	NaN	NaN	80.559998	224.199997	NaN	21.049999	63.2400
3	2003-03-01 01:00:00	NaN	2.45	NaN	NaN	NaN	78.370003	450.399994	NaN	4.220000	67.8399
4	2003-03-01 01:00:00	NaN	3.26	NaN	NaN	NaN	96.250000	479.100006	NaN	8.460000	95.7799
...	...	...	...	...	...	...	...	...	...	...	
243979	2003-10-01 00:00:00	0.20	0.16	2.01	3.17	0.02	31.799999	32.299999	1.68	34.049999	7.3800
243980	2003-10-01 00:00:00	0.32	0.08	0.36	0.72	NaN	10.450000	14.760000	1.00	34.610001	7.4000
243981	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	34.639999	50.810001	NaN	32.160000	16.8300
243982	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	32.580002	41.020000	NaN	NaN	13.5700
243983	2003-10-01 00:00:00	1.00	0.29	2.15	6.41	0.07	37.150002	56.849998	2.28	21.480000	12.3500

243984 rows × 16 columns

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

Out[3]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10	I
0	2003-03-01 01:00:00	NaN	1.72	NaN	NaN	NaN	73.900002	316.299988	NaN	10.550000	55.209999	I
1	2003-03-01 01:00:00	NaN	1.45	NaN	NaN	0.26	72.110001	250.000000	0.73	6.720000	52.389999	I
2	2003-03-01 01:00:00	NaN	1.57	NaN	NaN	NaN	80.559998	224.199997	NaN	21.049999	63.240002	I
3	2003-03-01 01:00:00	NaN	2.45	NaN	NaN	NaN	78.370003	450.399994	NaN	4.220000	67.839996	I
4	2003-03-01 01:00:00	NaN	3.26	NaN	NaN	NaN	96.250000	479.100006	NaN	8.460000	95.779999	I
5	2003-03-01 01:00:00	8.41	1.94	9.83	21.49	0.45	90.300003	384.899994	9.48	9.950000	95.150002	'
6	2003-03-01 01:00:00	NaN	1.38	NaN	NaN	0.29	89.580002	230.000000	NaN	7.200000	54.000000	I
7	2003-03-01 01:00:00	NaN	1.58	NaN	NaN	0.30	93.639999	334.600006	NaN	4.190000	26.620001	I
8	2003-03-01 01:00:00	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	I
9	2003-03-01 01:00:00	NaN	1.92	NaN	NaN	NaN	71.839996	181.399994	NaN	5.330000	39.360001	I

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

Out[4]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
243964	2003-10-01 00:00:00	NaN	0.63	NaN	NaN	NaN	49.740002	73.010002	NaN	15.310000	14.97
243965	2003-10-01 00:00:00	NaN	0.10	NaN	NaN	NaN	30.700001	38.430000	NaN	20.980000	4.83
243966	2003-10-01 00:00:00	NaN	0.16	NaN	NaN	0.12	43.160000	75.720001	NaN	24.850000	10.05
243967	2003-10-01 00:00:00	NaN	0.38	NaN	NaN	NaN	47.040001	88.839996	NaN	12.540000	11.60
243968	2003-10-01 00:00:00	NaN	0.14	NaN	NaN	0.15	23.430000	26.389999	NaN	27.730000	15.70
243969	2003-10-01 00:00:00	NaN	0.12	NaN	NaN	NaN	27.170000	30.570000	NaN	23.020000	10.48
243970	2003-10-01 00:00:00	0.68	0.43	1.12	NaN	0.00	55.169998	100.900002	NaN	14.590000	15.84
243971	2003-10-01 00:00:00	NaN	0.01	NaN	NaN	NaN	30.930000	39.430000	NaN	23.280001	13.31
243972	2003-10-01 00:00:00	NaN	0.24	NaN	NaN	NaN	36.599998	39.680000	NaN	24.700001	25.02
243973	2003-10-01 00:00:00	NaN	0.01	NaN	NaN	0.03	22.209999	25.650000	NaN	18.570000	7.78
243974	2003-10-01 00:00:00	NaN	0.28	NaN	NaN	NaN	23.790001	29.990000	NaN	25.540001	10.48
243975	2003-10-01 00:00:00	NaN	0.22	NaN	NaN	NaN	23.730000	27.709999	NaN	24.360001	10.31
243976	2003-10-01 00:00:00	NaN	0.15	NaN	NaN	NaN	37.270000	55.000000	NaN	20.980000	13.80
243977	2003-10-01 00:00:00	0.20	0.03	NaN	NaN	NaN	36.580002	43.090000	NaN	17.129999	7.61
243978	2003-10-01 00:00:00	NaN	0.24	NaN	NaN	0.06	25.129999	28.129999	NaN	30.660000	8.05
243979	2003-10-01 00:00:00	0.20	0.16	2.01	3.17	0.02	31.799999	32.299999	1.68	34.049999	7.38
243980	2003-10-01 00:00:00	0.32	0.08	0.36	0.72	NaN	10.450000	14.760000	1.00	34.610001	7.40

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	PM10
243981	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	34.639999	50.810001	NaN	32.160000	16.83
243982	2003-10-01 00:00:00	NaN	NaN	NaN	NaN	0.07	32.580002	41.020000	NaN	NaN	13.57
243983	2003-10-01 00:00:00	1.00	0.29	2.15	6.41	0.07	37.150002	56.849998	2.28	21.480000	12.35

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

Out[5]:

	BEN	CO	EBE	MXY	NMHC	NO_2
count	69745.000000	225340.000000	61244.000000	42045.000000	111951.000000	242625.000000
mean	2.106316	0.702223	2.448503	5.352770	0.153325	58.383284
std	2.386797	0.610948	3.061722	5.955294	0.145561	31.566000
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.700000	0.340000	0.890000	1.650000	0.070000	34.709999
50%	1.420000	0.540000	1.650000	3.580000	0.110000	54.779999
75%	2.650000	0.860000	3.010000	6.830000	0.190000	77.019997
max	66.389999	12.860000	162.199997	177.600006	4.360000	386.500000

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

Out[6]: (243984, 16)

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

Out[7]: 3903744

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	True	False	True	True	False	False	False	False	False	False	True	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
...	...	...	...	...	...	...	...	...	...	...	...	...	...
243979	False	False	False	False	False	False	False	False	False	False	False	False	False
243980	False	False	False	False	False	True	False	False	False	False	False	False	False
243981	False	True	True	True	True	False	False	False	True	False	False	True	False
243982	False	True	True	True	True	False	False	False	True	True	False	True	False
243983	False	False	False	False	False	False	False	False	False	False	False	False	False

243984 rows × 16 columns

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

Out[9]:

	date	BEN	CO	EBE	MXY	NMHC	NO_2	NOx	OXY	O_3	P
5	2003-03-01 01:00:00	8.41	1.94	9.83	21.49	0.45	90.300003	384.899994	9.48	9.950000	95.15
23	2003-03-01 01:00:00	3.46	1.27	3.43	7.08	0.18	54.250000	173.300003	3.37	6.540000	53.00
27	2003-03-01 01:00:00	6.39	1.79	5.75	10.88	0.33	75.459999	281.100006	3.68	6.690000	63.84
33	2003-03-01 02:00:00	7.42	1.47	10.63	24.73	0.35	83.309998	277.200012	11.00	9.900000	58.88
51	2003-03-01 02:00:00	3.62	1.29	3.20	7.08	0.19	42.209999	166.300003	3.41	6.380000	47.59
...	...	...	...	...	...	...	...	...	...	...	
243955	2003-09-30 23:00:00	1.75	0.41	3.07	9.38	0.09	46.290001	77.709999	3.11	18.280001	7.52
243957	2003-10-01 00:00:00	2.35	0.60	3.88	10.86	0.11	61.240002	133.100006	0.89	10.900000	10.24
243961	2003-10-01 00:00:00	2.97	0.82	4.53	10.88	0.05	36.529999	131.300003	5.52	12.940000	25.68
243979	2003-10-01 00:00:00	0.20	0.16	2.01	3.17	0.02	31.799999	32.299999	1.68	34.049999	7.38
243983	2003-10-01 00:00:00	1.00	0.29	2.15	6.41	0.07	37.150002	56.849998	2.28	21.480000	12.35

33010 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']]
```

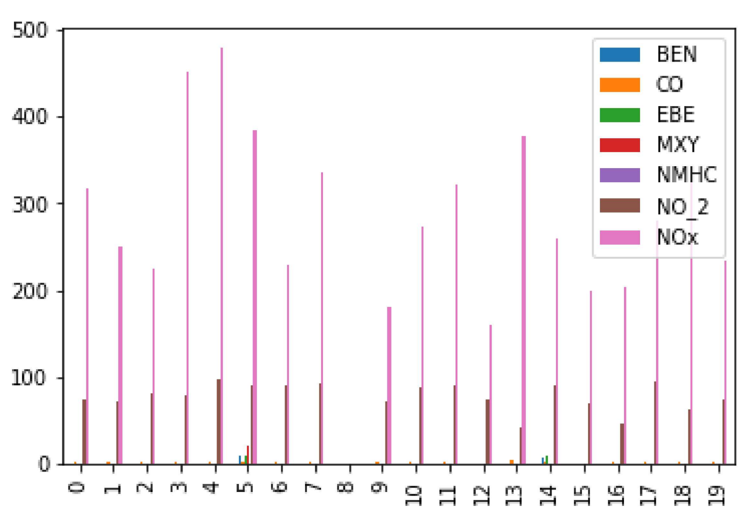
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	MXY	NMHC	NO_2	NOx
0	NaN	1.72	NaN	NaN	NaN	73.900002	316.299988
1	NaN	1.45	NaN	NaN	0.26	72.110001	250.000000
2	NaN	1.57	NaN	NaN	NaN	80.559998	224.199997
3	NaN	2.45	NaN	NaN	NaN	78.370003	450.399994
4	NaN	3.26	NaN	NaN	NaN	96.250000	479.100006
5	8.41	1.94	9.83	21.49	0.45	90.300003	384.899994
6	NaN	1.38	NaN	NaN	0.29	89.580002	230.000000
7	NaN	1.58	NaN	NaN	0.30	93.639999	334.600006
8	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9	NaN	1.92	NaN	NaN	NaN	71.839996	181.399994
10	NaN	1.33	NaN	NaN	0.31	87.919998	273.399994
11	NaN	2.18	NaN	NaN	NaN	89.849998	320.799988
12	NaN	1.14	NaN	NaN	0.25	73.870003	159.600006
13	NaN	4.68	NaN	NaN	NaN	42.189999	377.000000
14	6.97	1.44	10.27	NaN	0.47	91.010002	259.500000
15	NaN	1.25	NaN	NaN	NaN	70.570000	198.500000
16	NaN	1.64	NaN	NaN	NaN	45.470001	202.800003
17	NaN	1.85	NaN	NaN	0.59	94.510002	279.100006
18	NaN	1.74	NaN	NaN	NaN	62.259998	331.000000
19	NaN	1.54	NaN	NaN	NaN	73.239998	232.899994

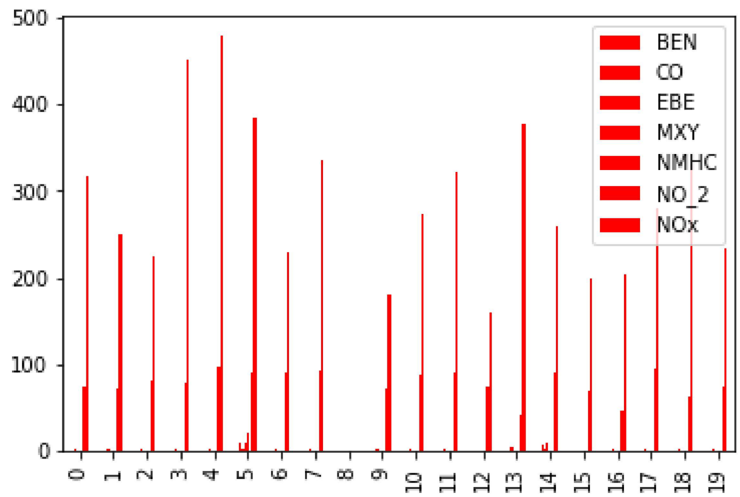
In [13]:
```
dd.plot.bar()
```

Out[13]: <AxesSubplot:>



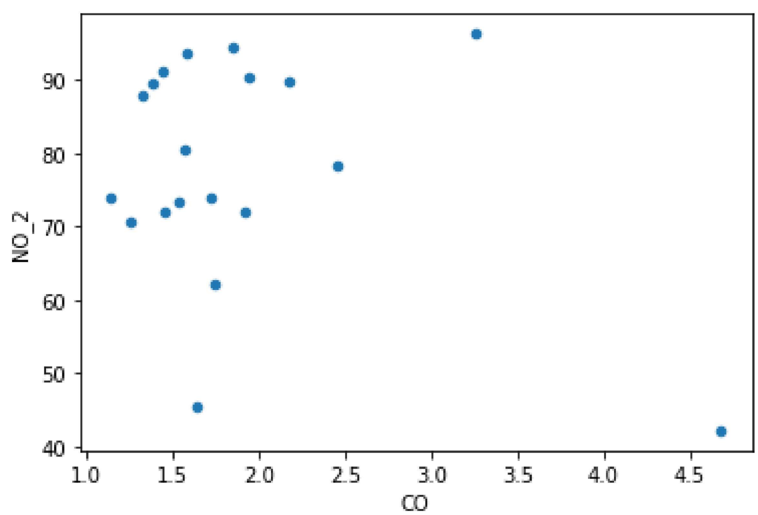
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



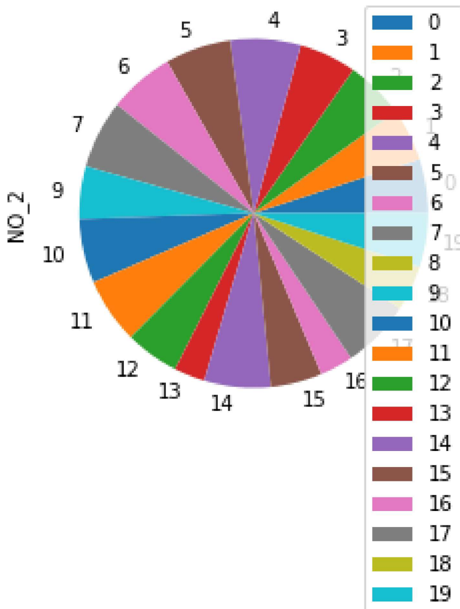
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

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



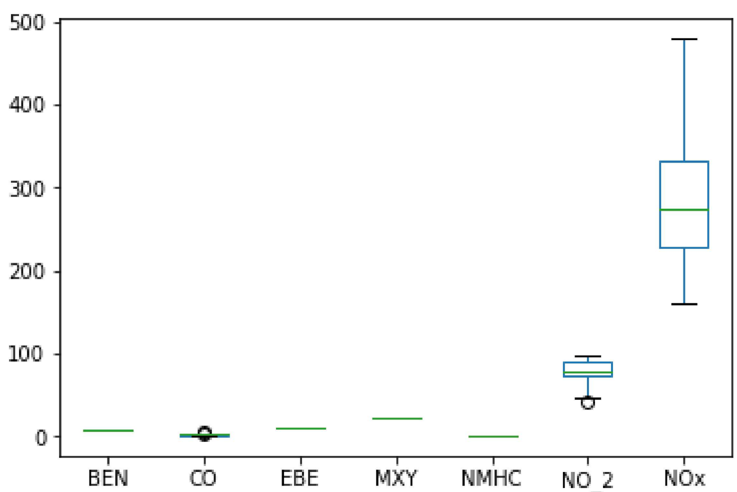
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



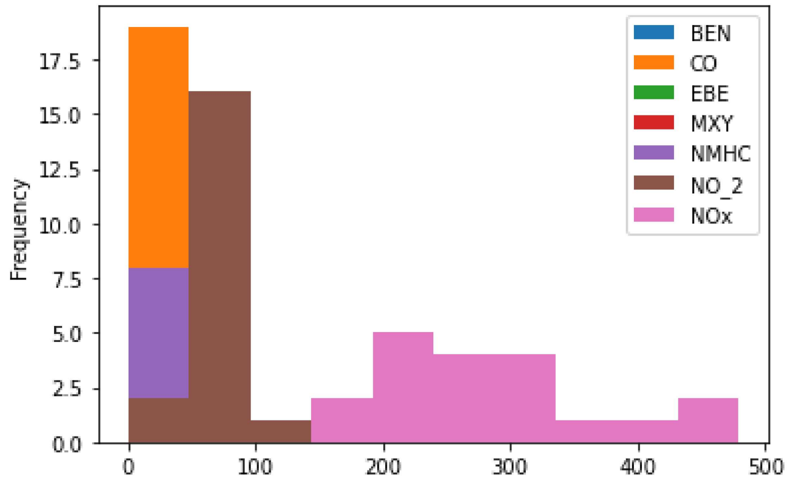
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



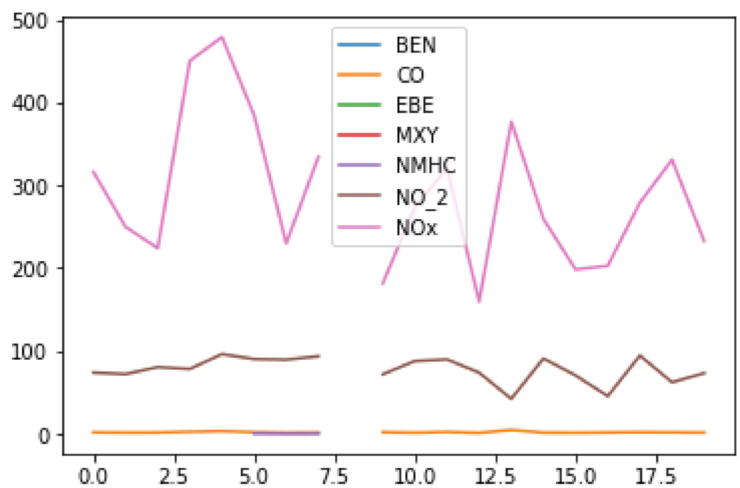
In [18]:
```
dd.plot.hist()
```

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



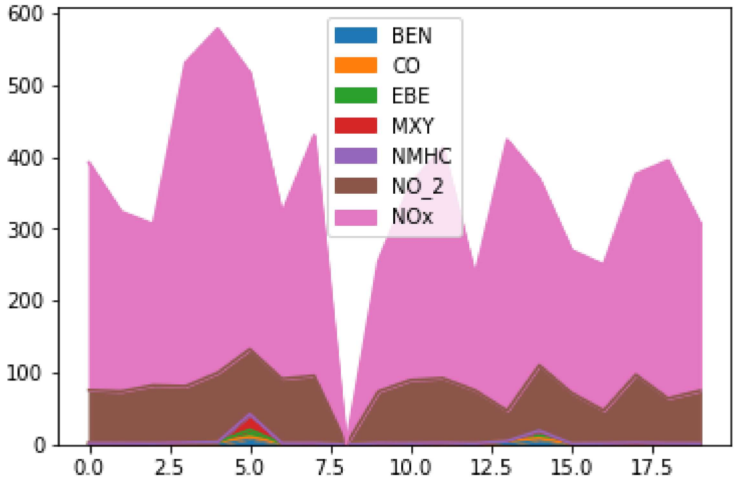
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



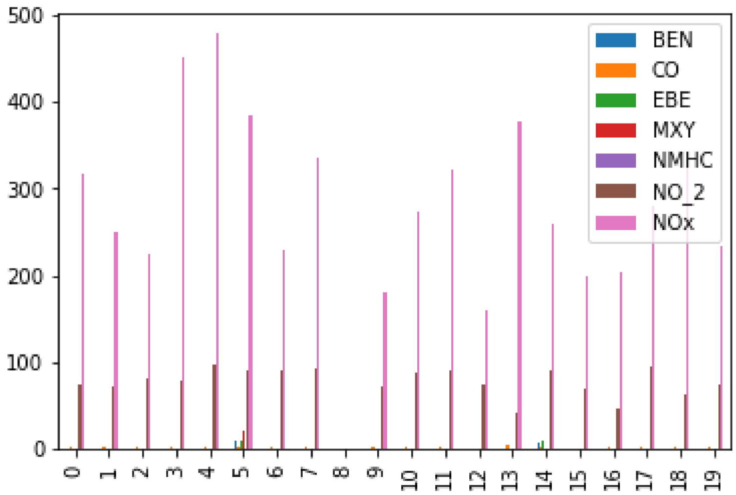
In [20]:
```
dd.plot.area()
```

Out[20]: <AxesSubplot:>



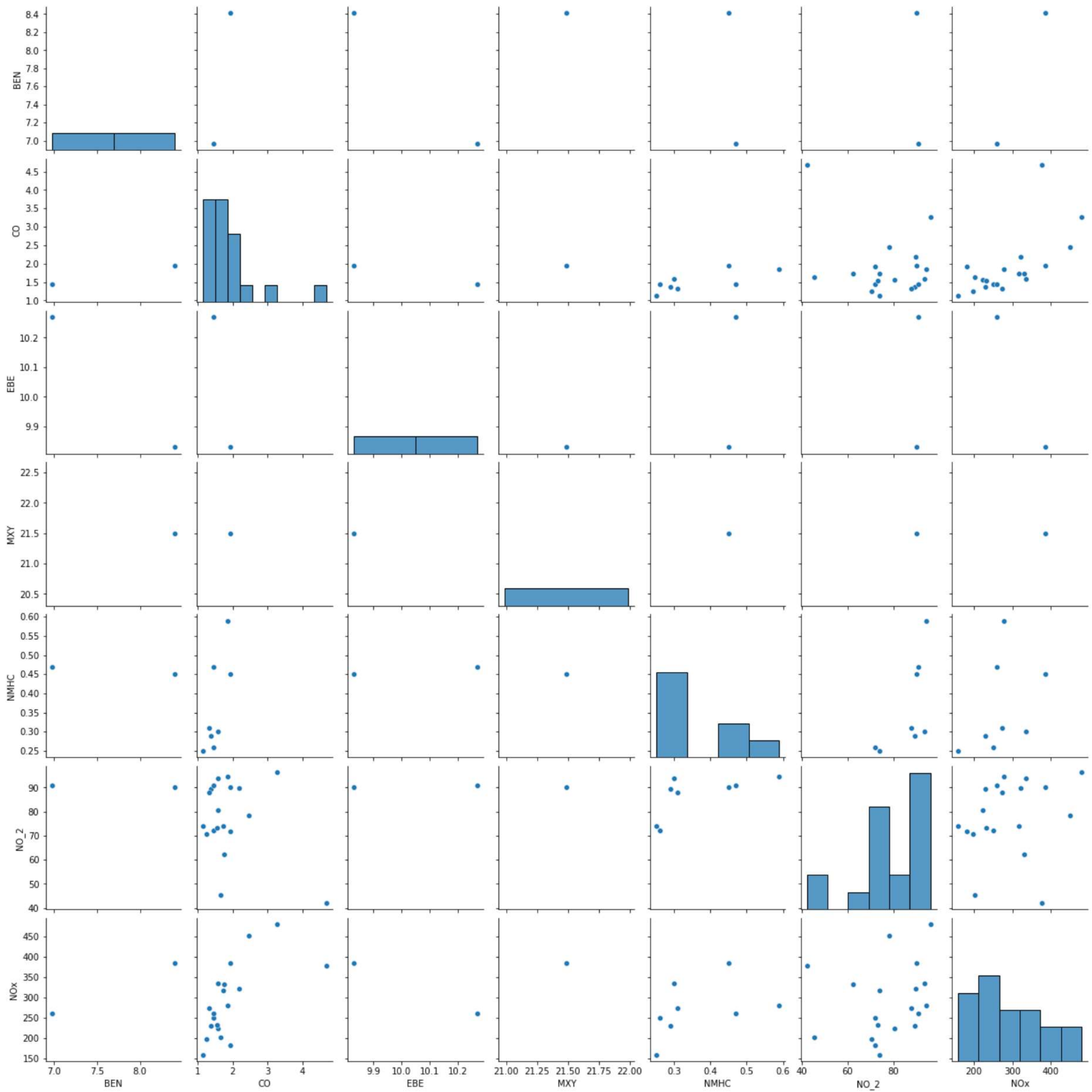
In [21]:
```
dd.plot.bar()
```

Out[21]: <AxesSubplot:>



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

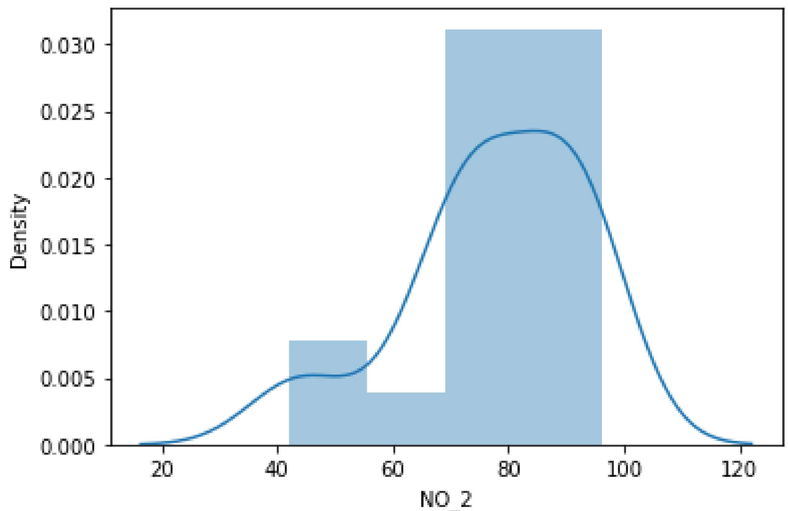
Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x227cdd74250>
```



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'>



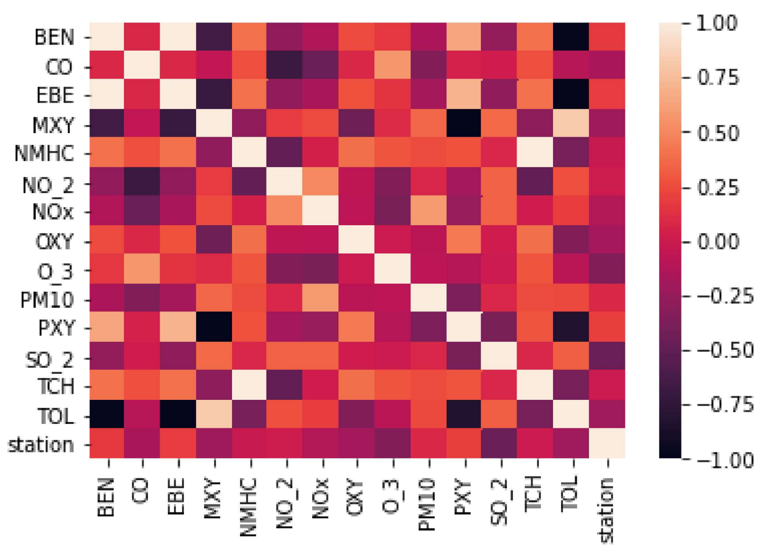
In [24]:
```
ds=data.fillna(20)
```

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:>



LinearRegression()

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_)
```

```
28079133.537731484
```

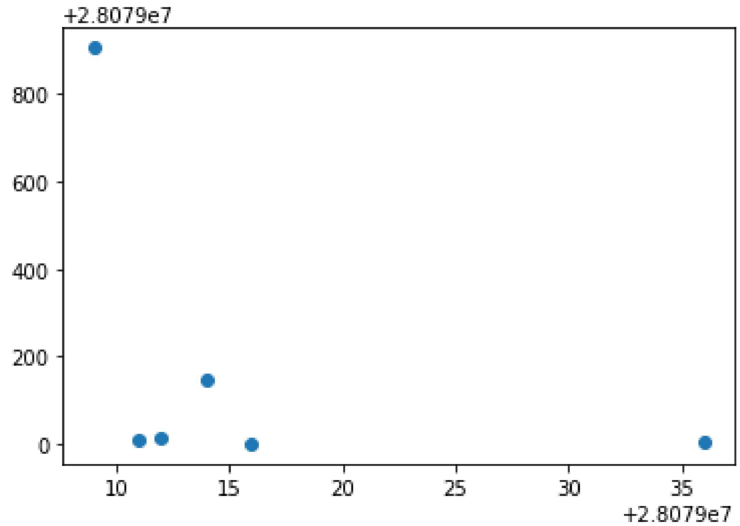
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	-2.594413
CO	45.277506
EBE	3.421176
MXY	-5.269043
NMHC	-1.418602
NO_2	-0.608823
NOx	-0.151573

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

Out[33]: <matplotlib.collections.PathCollection at 0x227d3f6f640>



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

```
-1669.7043953503141
```

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

Out[35]: -1669.7043953503141

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

Out[36]: 0.6565880695333406

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]: -12.087957480924334

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

Out[40]: 0.1689670989626525

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]:
```
-0.3910425400763189
```
In [43]:
```
la.score(x_train,y_train)
```
Out[43]:
```
0.0578739310630606
```

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_)
```
```
[ 0.13089663  3.25747742  0.65099835 -0.         -0.25047284 -0.09191952
 -0.03271323]
```
In [46]:
```
print(en.intercept_)
```
```
28079016.943015482
```
In [47]:
```
prediction=en.predict(x_test)
```
In [48]:
```
print(en.score(x_test,y_test))
```
```
-13.538946679247568
```

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[['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)
```

```
[28079039]
```

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

Out[59]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079017, 28079018, 28079019, 28079035, 28079036, 28079038,
       28079039, 28079040], dtype=int64)
```

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

Out[60]: 9.156433725894692e-05

RandomForestClassifier()

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')
```

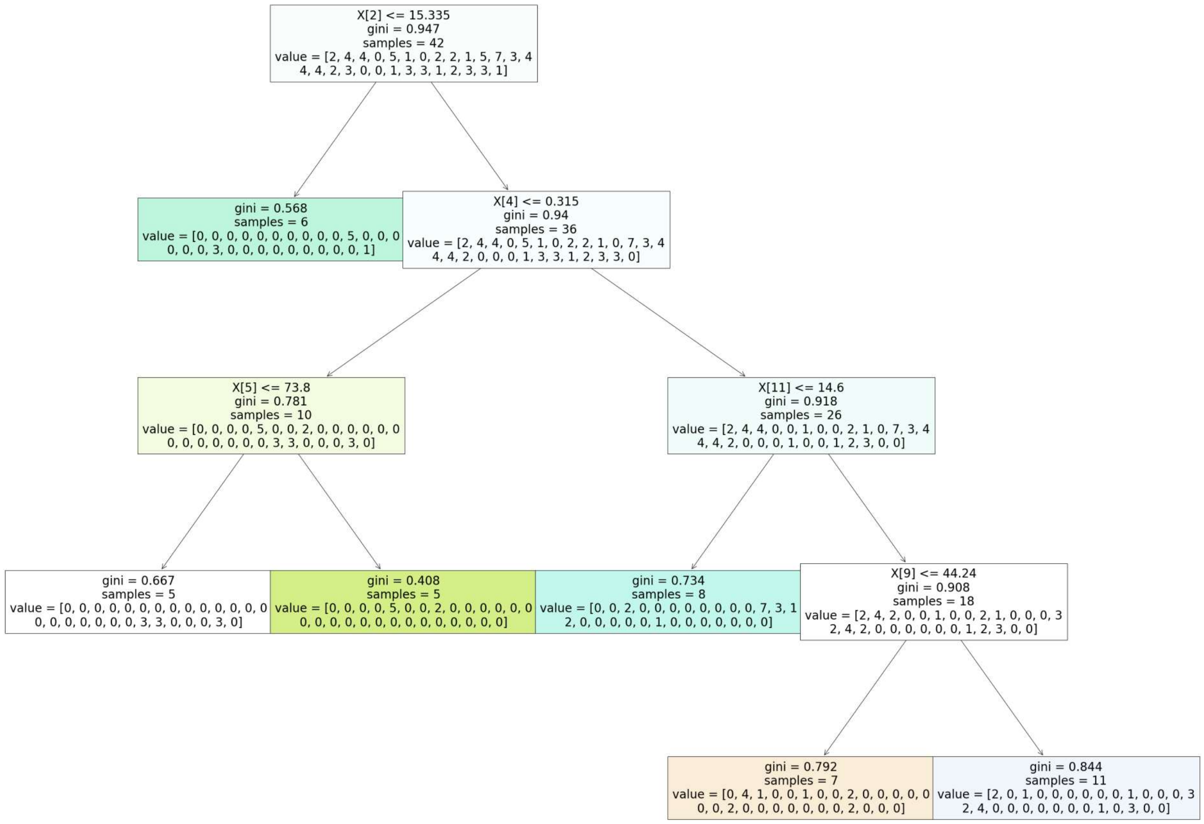
In [70]:
```
grid_search.best_score_
```

Out[70]:
```
0.4
```

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(930.0, 1956.96, 'X[2] <= 15.335\ngini = 0.947\nsamples = 42\nvalue =
[2, 4, 4, 0, 5, 1, 0, 2, 2, 1, 5, 7, 3, 4\n4, 4, 2, 3, 0, 0, 1, 3, 3, 1, 2,
3, 3, 1]'),
 Text(620.0, 1522.0800000000002, 'gini = 0.568\nsamples = 6\nvalue = [0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0\n0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1]'),
 Text(1240.0, 1522.0800000000002, 'X[4] <= 0.315\ngini = 0.94\nsamples = 36\n
value = [2, 4, 4, 0, 5, 1, 0, 2, 2, 1, 0, 7, 3, 4\n4, 4, 2, 0, 0, 0, 1, 3, 3,
1, 2, 3, 3, 0]'),
 Text(620.0, 1087.2, 'X[5] <= 73.8\ngini = 0.781\nsamples = 10\nvalue = [0,
0, 0, 0, 5, 0, 0, 2, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 3,
0]'),
 Text(310.0, 652.3200000000002, 'gini = 0.667\nsamples = 5\nvalue = [0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 3, 0]'),
 Text(930.0, 652.3200000000002, 'gini = 0.408\nsamples = 5\nvalue = [0, 0, 0,
0, 5, 0, 0, 2, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1860.0, 1087.2, 'X[11] <= 14.6\ngini = 0.918\nsamples = 26\nvalue = [2,
4, 4, 0, 0, 1, 0, 0, 2, 1, 0, 7, 3, 4\n4, 4, 2, 0, 0, 0, 1, 0, 0, 1, 2, 3, 0,
0]'),
 Text(1550.0, 652.3200000000002, 'gini = 0.734\nsamples = 8\nvalue = [0, 0,
2, 0, 0, 0, 0, 0, 0, 0, 0, 7, 3, 1\n2, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0]'),
 Text(2170.0, 652.3200000000002, 'X[9] <= 44.24\ngini = 0.908\nsamples = 18\n
value = [2, 4, 2, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 3\n2, 4, 2, 0, 0, 0, 0, 0, 0,
1, 2, 3, 0, 0]'),
 Text(1860.0, 217.44000000000005, 'gini = 0.792\nsamples = 7\nvalue = [0, 4,
1, 0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 0\n0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0,
0]'),
 Text(2480.0, 217.44000000000005, 'gini = 0.844\nsamples = 11\nvalue = [2, 0,
1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 3\n2, 4, 0, 0, 0, 0, 0, 0, 0, 1, 0, 3, 0,
0]')]
```



Conclusion : ElasticNet() 28079016.943015482 HIGH RANGE

In [ ]:

```

```