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_2016.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
0	2016-11-01 01:00:00	NaN	0.7	NaN	NaN	153.0	77.0	NaN	NaN	NaN	7.0	NaN	NaN	2
1	2016-11-01 01:00:00	3.1	1.1	2.0	0.53	260.0	144.0	4.0	46.0	24.0	18.0	2.44	14.4	2
2	2016-11-01 01:00:00	5.9	NaN	7.5	NaN	297.0	139.0	NaN	NaN	NaN	NaN	NaN	26.0	2
3	2016-11-01 01:00:00	NaN	1.0	NaN	NaN	154.0	113.0	2.0	NaN	NaN	NaN	NaN	NaN	2
4	2016-11-01 01:00:00	NaN	NaN	NaN	NaN	275.0	127.0	2.0	NaN	NaN	18.0	NaN	NaN	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
209491	2016-07-01 00:00:00	NaN	0.2	NaN	NaN	2.0	29.0	73.0	NaN	NaN	NaN	NaN	NaN	2
209492	2016-07-01 00:00:00	NaN	0.3	NaN	NaN	1.0	29.0	NaN	36.0	NaN	5.0	NaN	NaN	2
209493	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	1.0	19.0	71.0	NaN	NaN	NaN	NaN	NaN	2
209494	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	6.0	17.0	85.0	NaN	NaN	NaN	NaN	NaN	2
209495	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	2.0	46.0	61.0	34.0	NaN	NaN	NaN	NaN	2

209496 rows × 14 columns

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

Out[3]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	stati
0	2016-11-01 01:00:00	NaN	0.7	NaN	NaN	153.0	77.0	NaN	NaN	NaN	7.0	NaN	NaN	280790
1	2016-11-01 01:00:00	3.1	1.1	2.0	0.53	260.0	144.0	4.0	46.0	24.0	18.0	2.44	14.4	280790
2	2016-11-01 01:00:00	5.9	NaN	7.5	NaN	297.0	139.0	NaN	NaN	NaN	NaN	NaN	26.0	280790
3	2016-11-01 01:00:00	NaN	1.0	NaN	NaN	154.0	113.0	2.0	NaN	NaN	NaN	NaN	NaN	280790
4	2016-11-01 01:00:00	NaN	NaN	NaN	NaN	275.0	127.0	2.0	NaN	NaN	18.0	NaN	NaN	280790
5	2016-11-01 01:00:00	0.9	0.5	0.5	NaN	66.0	82.0	1.0	27.0	NaN	8.0	NaN	6.0	280790
6	2016-11-01 01:00:00	0.7	0.8	0.4	0.13	57.0	66.0	3.0	23.0	15.0	4.0	1.35	5.0	280790
7	2016-11-01 01:00:00	NaN	NaN	NaN	NaN	52.0	78.0	1.0	NaN	NaN	NaN	NaN	NaN	280790
8	2016-11-01 01:00:00	NaN	1.2	NaN	NaN	205.0	85.0	6.0	NaN	NaN	NaN	NaN	NaN	280790
9	2016-11-01 01:00:00	NaN	0.7	NaN	NaN	114.0	91.0	NaN	37.0	NaN	6.0	NaN	NaN	280790

In [4]:

```
data.tail(20)
```

Out[4]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
209476	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	4.0	24.0	77.0	NaN	NaN	8.0	NaN	NaN	28
209477	2016-07-01 00:00:00	0.2	0.3	0.1	NaN	2.0	28.0	83.0	33.0	NaN	4.0	NaN	0.7	28
209478	2016-07-01 00:00:00	0.1	0.2	0.1	0.02	1.0	6.0	89.0	16.0	9.0	2.0	1.15	0.2	28
209479	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	1.0	24.0	73.0	NaN	NaN	NaN	NaN	NaN	28
209480	2016-07-01 00:00:00	NaN	0.3	NaN	NaN	1.0	27.0	47.0	NaN	NaN	19.0	NaN	NaN	28
209481	2016-07-01 00:00:00	NaN	0.2	NaN	NaN	1.0	32.0	NaN	37.0	NaN	6.0	NaN	NaN	28
209482	2016-07-01 00:00:00	0.1	NaN	0.1	NaN	8.0	28.0	NaN	19.0	9.0	2.0	NaN	1.1	28
209483	2016-07-01 00:00:00	NaN	0.3	NaN	NaN	5.0	41.0	59.0	NaN	NaN	NaN	NaN	NaN	28
209484	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	4.0	31.0	NaN	30.0	NaN	5.0	NaN	NaN	28
209485	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	3.0	20.0	NaN	20.0	15.0	NaN	NaN	NaN	28
209486	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	2.0	34.0	NaN	21.0	14.0	NaN	NaN	NaN	28
209487	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	1.0	20.0	74.0	NaN	NaN	NaN	NaN	NaN	28
209488	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	28.0	54.0	NaN	21.0	13.0	NaN	NaN	NaN	28
209489	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	4.0	31.0	73.0	NaN	NaN	NaN	NaN	NaN	28
209490	2016-07-01 00:00:00	0.3	NaN	0.2	0.10	1.0	30.0	NaN	45.0	NaN	NaN	1.19	2.0	28
209491	2016-07-01 00:00:00	NaN	0.2	NaN	NaN	2.0	29.0	73.0	NaN	NaN	NaN	NaN	NaN	28
209492	2016-07-01 00:00:00	NaN	0.3	NaN	NaN	1.0	29.0	NaN	36.0	NaN	5.0	NaN	NaN	28

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
209493	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	1.0	19.0	71.0	NaN	NaN	NaN	NaN	NaN	28
209494	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	6.0	17.0	85.0	NaN	NaN	NaN	NaN	NaN	28
209495	2016-07-01 00:00:00	NaN	NaN	NaN	NaN	2.0	46.0	61.0	34.0	NaN	NaN	NaN	NaN	28

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

Out[5]:

	BEN	CO	EBE	NMHC	NO	NO_2	
count	50755.000000	85999.000000	50335.000000	25970.000000	208614.000000	208614.000000	1:
mean	0.632450	0.354954	0.374407	0.124340	22.058280	38.559248	
std	0.857079	0.234792	0.684403	0.117454	46.518291	28.970459	
min	0.100000	0.100000	0.100000	0.000000	1.000000	1.000000	
25%	0.100000	0.200000	0.100000	0.070000	2.000000	17.000000	
50%	0.300000	0.300000	0.200000	0.110000	6.000000	32.000000	
75%	0.700000	0.400000	0.400000	0.150000	19.000000	54.000000	
max	21.400000	4.500000	27.400000	9.070000	957.000000	324.000000	

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

Out[6]: (209496, 14)

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

Out[7]: 2932944

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

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL
0	False	True	False	True	True	False	False	True	True	True	False	True	True
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	True	False	True	False	False	True	True	True	True	True	False
3	False	True	False	True	True	False	False	False	True	True	True	True	True
4	False	True	True	True	True	False	False	False	True	True	False	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...
209491	False	True	False	True	True	False	False	False	True	True	True	True	True
209492	False	True	False	True	True	False	False	True	False	True	False	True	True
209493	False	True	True	True	True	False	False	False	True	True	True	True	True
209494	False	True	True	True	True	False	False	False	True	True	True	True	True
209495	False	True	True	True	True	False	False	False	False	True	True	True	True

209496 rows × 14 columns

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

Out[9]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
1	2016-11-01 01:00:00	3.1	1.1	2.0	0.53	260.0	144.0	4.0	46.0	24.0	18.0	2.44	14.4	28
6	2016-11-01 01:00:00	0.7	0.8	0.4	0.13	57.0	66.0	3.0	23.0	15.0	4.0	1.35	5.0	28
25	2016-11-01 02:00:00	2.7	1.0	2.1	0.40	139.0	114.0	4.0	37.0	21.0	14.0	2.30	15.0	28
30	2016-11-01 02:00:00	0.7	0.7	0.4	0.13	48.0	59.0	3.0	23.0	15.0	3.0	1.35	5.0	28
49	2016-11-01 03:00:00	1.7	0.8	1.4	0.25	53.0	90.0	4.0	31.0	19.0	10.0	1.95	10.7	28
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
209430	2016-06-30 22:00:00	0.1	0.2	0.1	0.02	1.0	5.0	97.0	19.0	12.0	2.0	1.15	0.2	28
209449	2016-06-30 23:00:00	0.6	0.4	0.3	0.15	14.0	63.0	54.0	29.0	13.0	16.0	1.48	1.9	28
209454	2016-06-30 23:00:00	0.1	0.2	0.1	0.02	1.0	7.0	91.0	16.0	9.0	2.0	1.15	0.3	28
209473	2016-07-01 00:00:00	0.6	0.4	0.3	0.16	11.0	68.0	45.0	24.0	14.0	16.0	1.50	1.9	28
209478	2016-07-01 00:00:00	0.1	0.2	0.1	0.02	1.0	6.0	89.0	16.0	9.0	2.0	1.15	0.2	28

16932 rows × 14 columns

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

Out[10]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'NMHC', 'NO', 'NO_2', 'O_3', 'PM10', 'PM2
5',
       'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

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

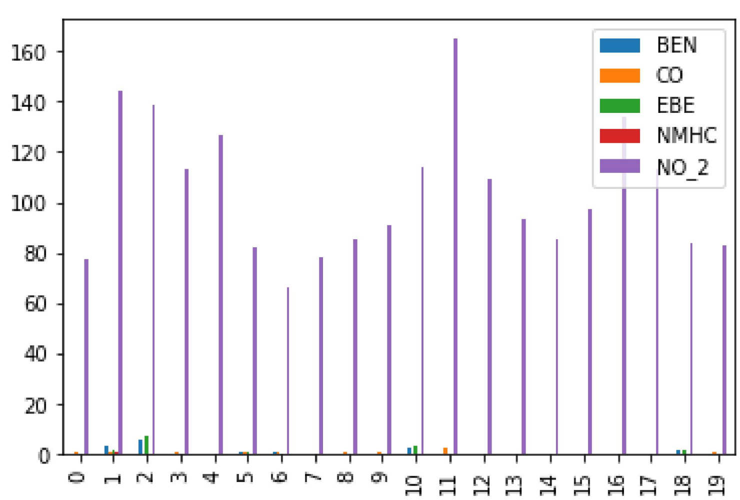
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

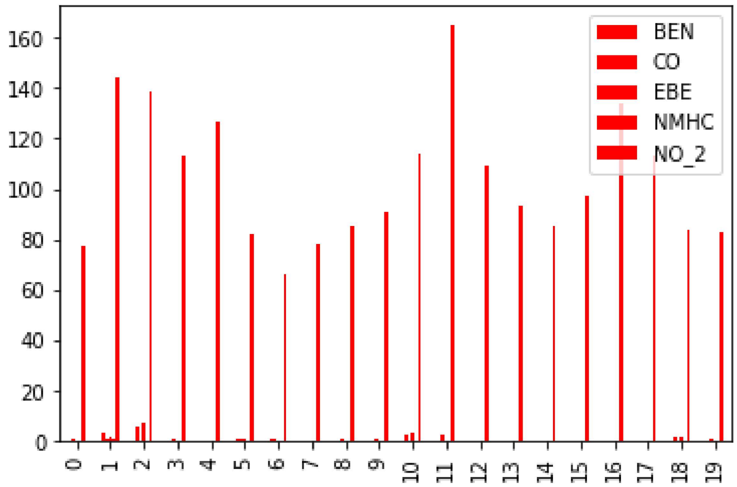
	BEN	CO	EBE	NMHC	NO_2
0	NaN	0.7	NaN	NaN	77.0
1	3.1	1.1	2.0	0.53	144.0
2	5.9	NaN	7.5	NaN	139.0
3	NaN	1.0	NaN	NaN	113.0
4	NaN	NaN	NaN	NaN	127.0
5	0.9	0.5	0.5	NaN	82.0
6	0.7	0.8	0.4	0.13	66.0
7	NaN	NaN	NaN	NaN	78.0
8	NaN	1.2	NaN	NaN	85.0
9	NaN	0.7	NaN	NaN	91.0
10	2.5	NaN	3.3	NaN	114.0
11	NaN	2.4	NaN	NaN	165.0
12	NaN	NaN	NaN	NaN	109.0
13	NaN	NaN	NaN	NaN	93.0
14	NaN	NaN	NaN	NaN	85.0
15	NaN	NaN	NaN	NaN	97.0
16	NaN	NaN	NaN	NaN	134.0
17	NaN	NaN	NaN	NaN	113.0
18	1.4	NaN	1.3	0.20	84.0
19	NaN	0.5	NaN	NaN	83.0

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

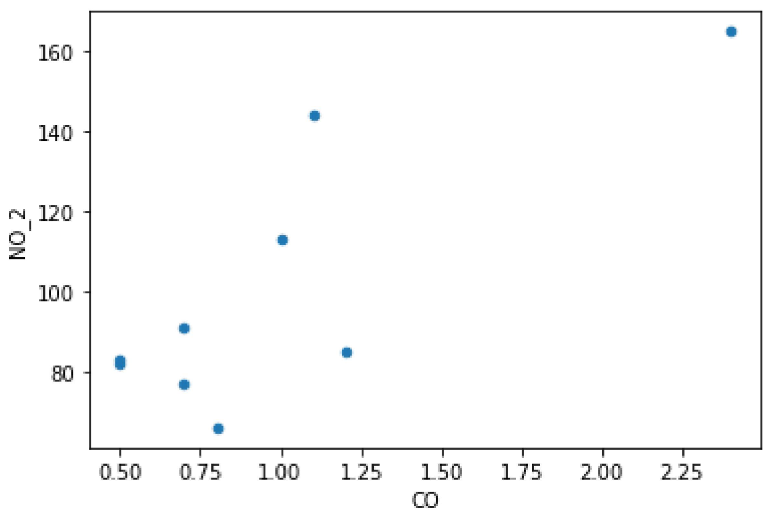
Out[13]: <AxesSubplot:>



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

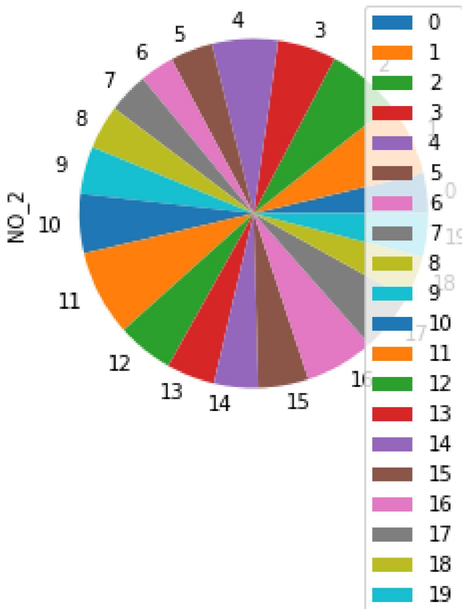


In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```
Out[15]: <AxesSubplot:xlabel='CO', ylabel='NO_2'>



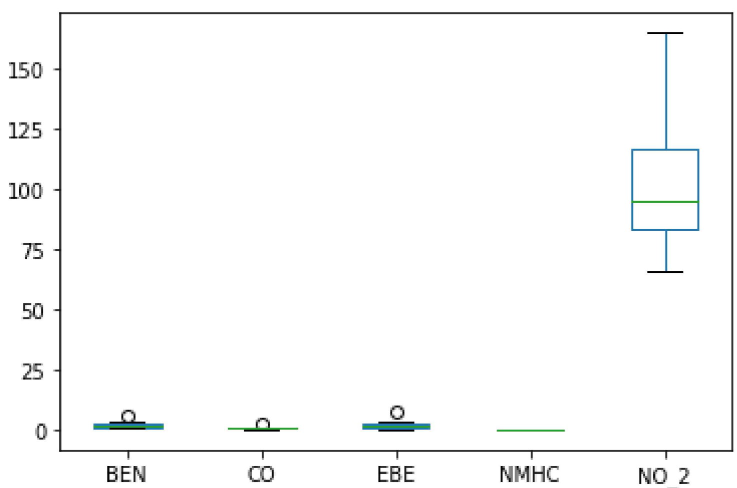
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



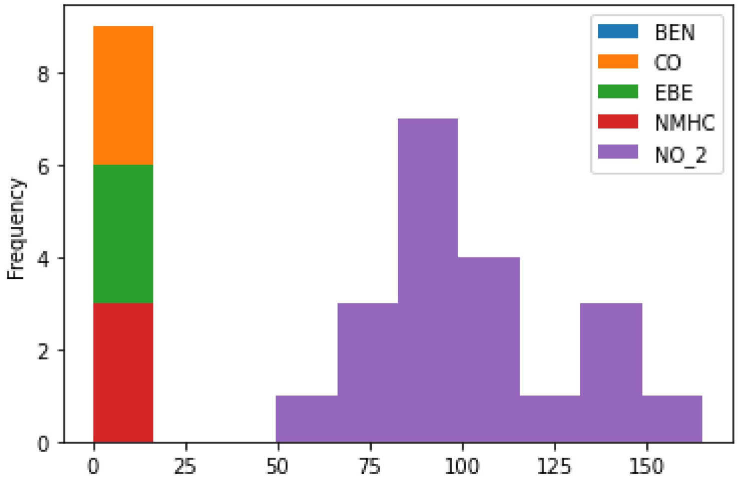
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



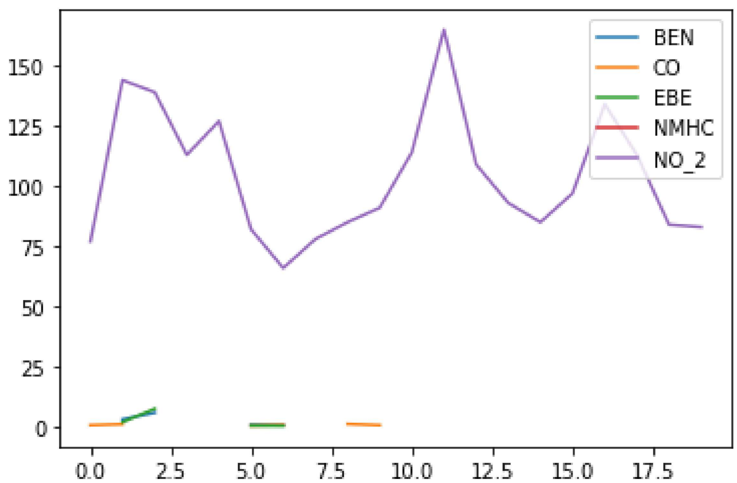
In [18]:
```
dd.plot.hist()
```

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



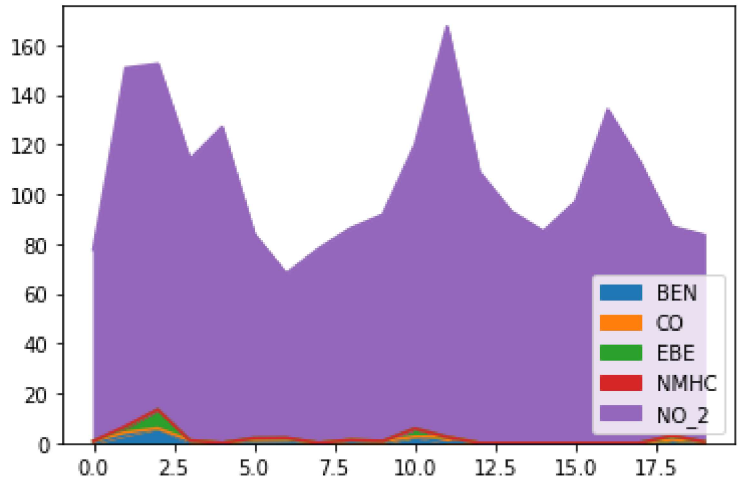
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



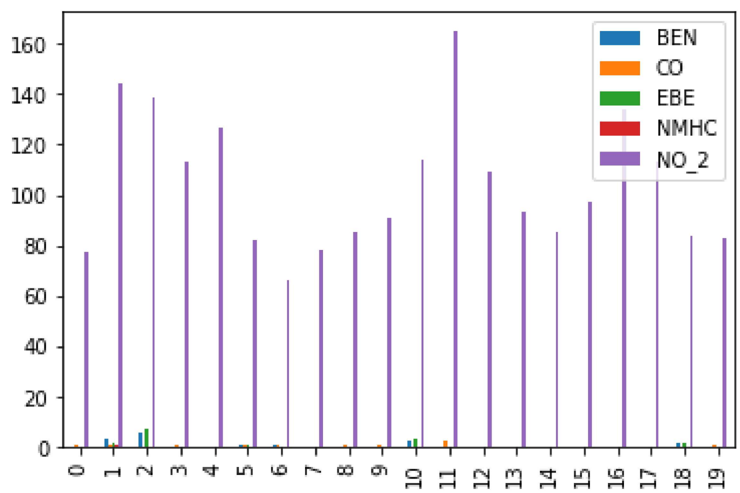
```
In [20]: dd.plot.area()
```

Out[20]: <AxesSubplot:>



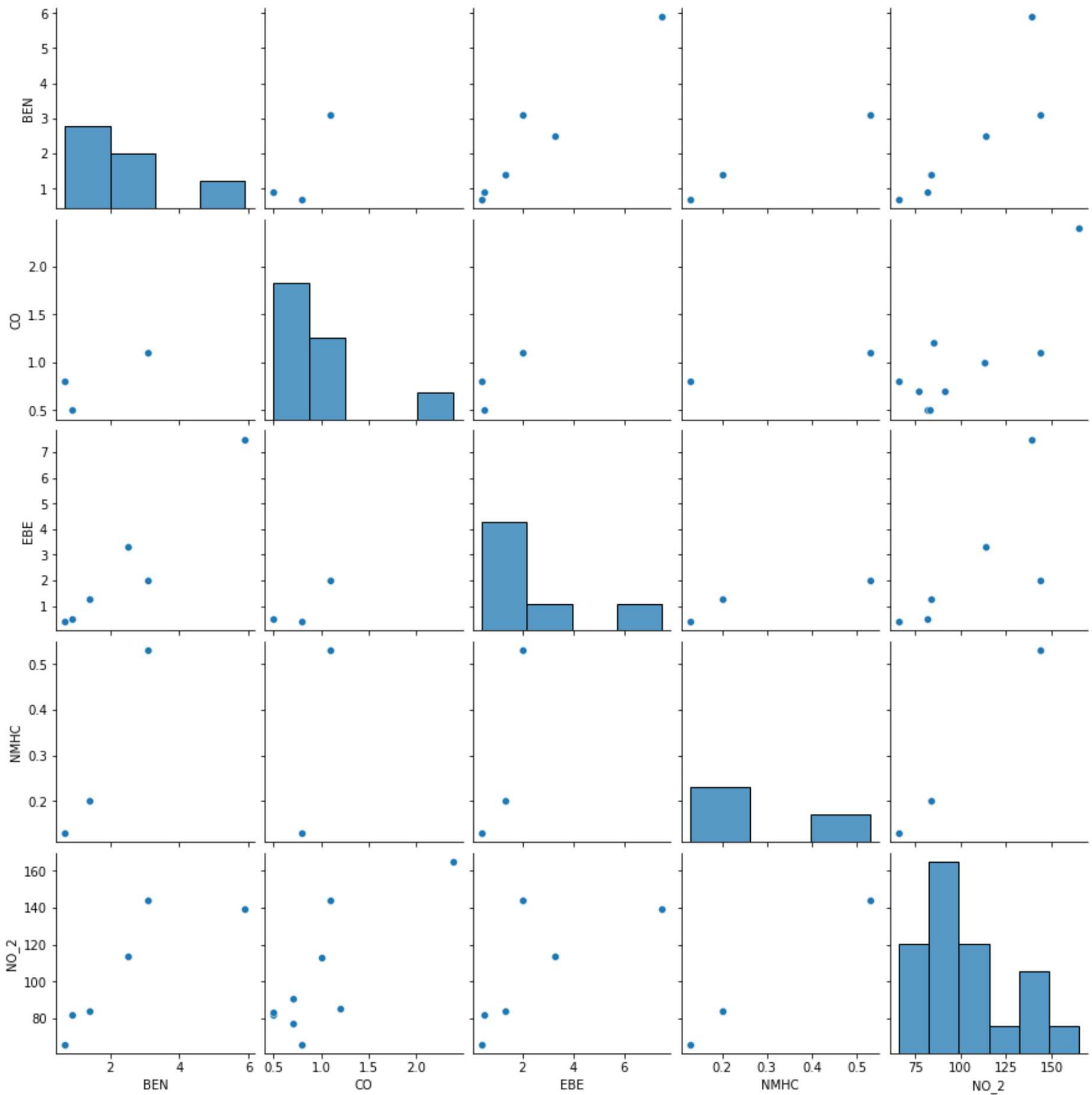

```
In [21]: dd.plot.bar()
```

Out[21]: <AxesSubplot:>



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

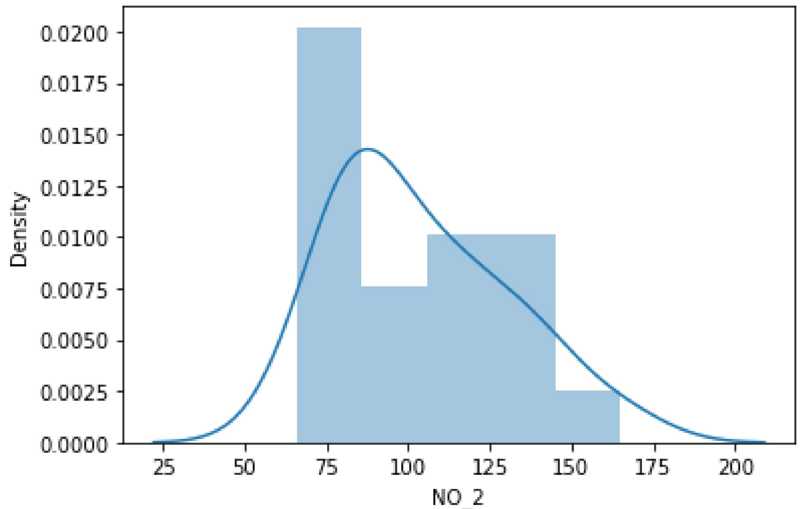
Out[22]: <seaborn.axisgrid.PairGrid at 0x1caa2747c70>



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'>



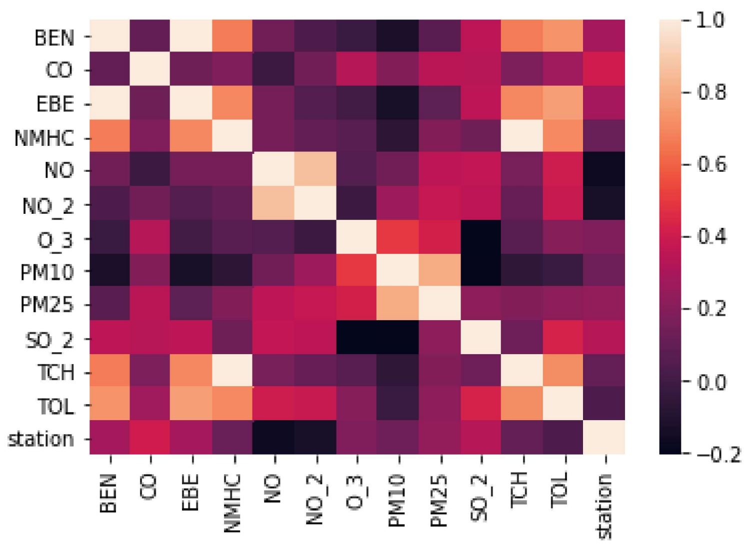
In [24]:
```
ds=data.fillna(20)
```

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

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

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

Out[27]: <AxesSubplot:>



In [28]:
```
x= ssd[['BEN','CO', 'EBE','NMHC', 'NO_2']]
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_)
```

28079032.39849289

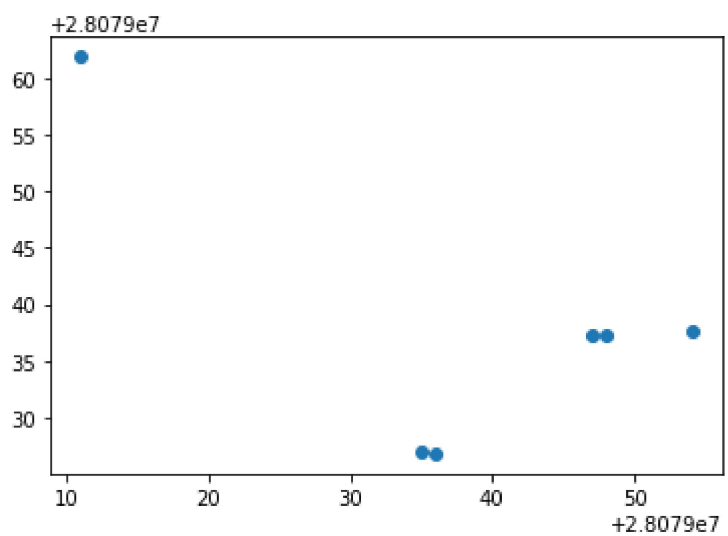
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	-18.062999
CO	0.543611
EBE	18.459225
NMHC	-0.762618
NO_2	0.014609

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

Out[33]: <matplotlib.collections.PathCollection at 0x1caa541eee0>



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

```
-1.736951953080799
```

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

Out[35]:
```
-1.736951953080799
```

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

Out[36]:
```
0.31143336104469654
```

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]:
```
-0.0609562394769476l6
```

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

Out[40]:
```
0.23910889006323122
```

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]:
```
-0.19039613318274196
```

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

Out[43]:
```
0.22148355450604817
```

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_)
```

```
[-0.11200587  0.82311176  0.31603659 -0.15822879 -0.02180312]
```

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

```
28079024.923744354
```

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

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

```
-0.025875730398854824
```

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','NMHC', 'NO_2']]
target_vector=ssd['station']
```

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

Out[52]:
```
(20, 5)
```

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]]
```

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

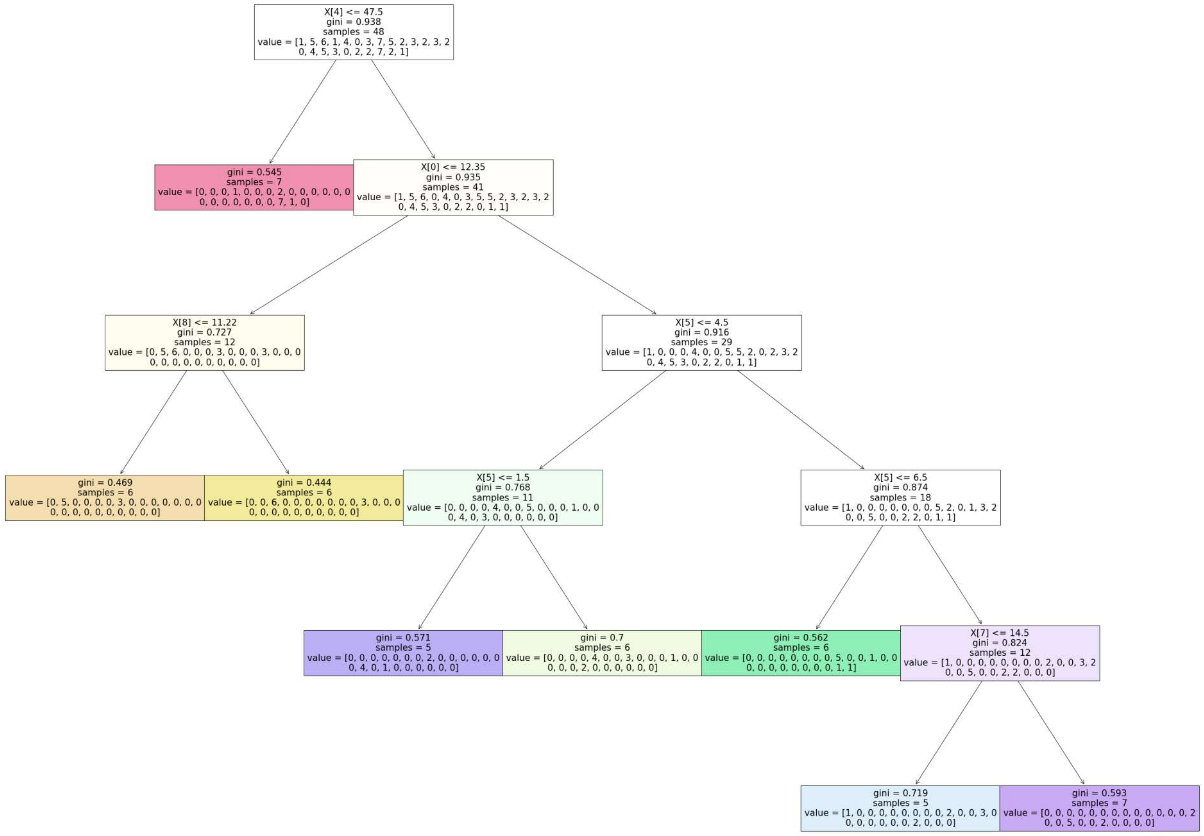
```
[28079039]
```

In [59]:
```
logr.classes_
```
Out[59]:
```
array([28079004, 28079008, 28079011, 28079016, 28079017, 28079018,
       28079024, 28079027, 28079035, 28079036, 28079038, 28079039,
       28079040, 28079047, 28079048, 28079049, 28079050, 28079054,
       28079055, 28079056], dtype=int64)
```
In [60]:
```
logr.predict_proba(observation)[0][0]
```
Out[60]:
```
6.359675284065992e-05
```
In [61]:
```
ged=data[['BEN','CO','EBE','NMHC','NO_2','O_3','PM10','SO_2','TCH','TOL','stati
```
In [62]:
```
d=ged.fillna(20)
```
In [63]:
```
dg=d.head(100)
```
In [64]:
```
x=dg[['BEN','CO','EBE','NMHC','NO_2','O_3','PM10','SO_2','TCH','TOL']]
y=dg['station']
```
In [65]:
```
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 [66]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```
Out[66]:
```
RandomForestClassifier()
```
In [67]:
```
parameters = {'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 [68]:
```
from sklearn.model_selection import GridSearchCV
grid_search= GridSearchCV(estimator = rfc,param_grid=parameters,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[68]:
```
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 [69]:
```
grid_search.best_score_
```
Out[69]:
```
0.41428571428571426
```

In [70]:
```
rfc_best=grid_search.best_estimator_
```

In [71]:
```python
from sklearn.tree import plot_tree
plt.figure(figsize=(50,40))
plot_tree(rfc_best.estimators_[5],filled=True)
```

Out[71]:
```
[Text(813.75, 1993.2, 'X[4] <= 47.5\ngini = 0.938\nsamples = 48\nvalue = [1, 
5, 6, 1, 4, 0, 3, 7, 5, 2, 3, 2, 3, 2\n0, 4, 5, 3, 0, 2, 2, 7, 2, 1]'),
 Text(581.25, 1630.8000000000002, 'gini = 0.545\nsamples = 7\nvalue = [0, 0, 
0, 1, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 7, 1, 0]'),
 Text(1046.25, 1630.8000000000002, 'X[0] <= 12.35\ngini = 0.935\nsamples = 41
\nvalue = [1, 5, 6, 0, 4, 0, 3, 5, 5, 2, 3, 2, 3, 2\n0, 4, 5, 3, 0, 2, 2, 0, 
1, 1]'),
 Text(465.0, 1268.4, 'X[8] <= 11.22\ngini = 0.727\nsamples = 12\nvalue = [0, 
5, 6, 0, 0, 0, 3, 0, 0, 0, 3, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(232.5, 906.0, 'gini = 0.469\nsamples = 6\nvalue = [0, 5, 0, 0, 0, 0, 3, 
0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(697.5, 906.0, 'gini = 0.444\nsamples = 6\nvalue = [0, 0, 6, 0, 0, 0, 0, 
0, 0, 0, 3, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1627.5, 1268.4, 'X[5] <= 4.5\ngini = 0.916\nsamples = 29\nvalue = [1, 
0, 0, 0, 4, 0, 0, 5, 5, 2, 0, 2, 3, 2\n0, 4, 5, 3, 0, 2, 2, 0, 1, 1]'),
 Text(1162.5, 906.0, 'X[5] <= 1.5\ngini = 0.768\nsamples = 11\nvalue = [0, 0, 
0, 0, 4, 0, 0, 5, 0, 0, 0, 1, 0, 0\n0, 4, 0, 3, 0, 0, 0, 0, 0, 0]'),
 Text(930.0, 543.5999999999999, 'gini = 0.571\nsamples = 5\nvalue = [0, 0, 0, 
0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0\n0, 4, 0, 1, 0, 0, 0, 0, 0, 0]'),
 Text(1395.0, 543.5999999999999, 'gini = 0.7\nsamples = 6\nvalue = [0, 0, 0, 
0, 4, 0, 0, 3, 0, 0, 0, 1, 0, 0\n0, 0, 0, 2, 0, 0, 0, 0, 0, 0]'),
 Text(2092.5, 906.0, 'X[5] <= 6.5\ngini = 0.874\nsamples = 18\nvalue = [1, 0, 
0, 0, 0, 0, 0, 0, 5, 2, 0, 1, 3, 2\n0, 0, 5, 0, 0, 2, 2, 0, 1, 1]'),
 Text(1860.0, 543.5999999999999, 'gini = 0.562\nsamples = 6\nvalue = [0, 0, 
0, 0, 0, 0, 0, 0, 5, 0, 0, 1, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 1, 1]'),
 Text(2325.0, 543.5999999999999, 'X[7] <= 14.5\ngini = 0.824\nsamples = 12\nv
alue = [1, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 3, 2\n0, 0, 5, 0, 0, 2, 2, 0, 0, 
0]'),
 Text(2092.5, 181.19999999999982, 'gini = 0.719\nsamples = 5\nvalue = [1, 0, 
0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 3, 0\n0, 0, 0, 0, 0, 0, 2, 0, 0, 0]'),
 Text(2557.5, 181.19999999999982, 'gini = 0.593\nsamples = 7\nvalue = [0, 0, 
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2\n0, 0, 5, 0, 0, 2, 0, 0, 0, 0]')]
```



Conclusion : LogisticRegression() [28079039] HIGH RANGE

In [ ]: