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_2015.csv")
data
```

Out[2]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
0	2015-10-01 01:00:00	NaN	0.8	NaN	NaN	90.0	82.0	NaN	NaN	NaN	10.0	NaN	NaN	28
1	2015-10-01 01:00:00	2.0	0.8	1.6	0.33	40.0	95.0	4.0	37.0	24.0	12.0	1.83	8.3	28
2	2015-10-01 01:00:00	3.1	NaN	1.8	NaN	29.0	97.0	NaN	NaN	NaN	NaN	NaN	7.1	28
3	2015-10-01 01:00:00	NaN	0.6	NaN	NaN	30.0	103.0	2.0	NaN	NaN	NaN	NaN	NaN	28
4	2015-10-01 01:00:00	NaN	NaN	NaN	NaN	95.0	96.0	2.0	NaN	NaN	9.0	NaN	NaN	28
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210091	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	11.0	33.0	53.0	NaN	NaN	NaN	NaN	NaN	28
210092	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	1.0	5.0	NaN	26.0	NaN	10.0	NaN	NaN	28
210093	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	7.0	74.0	NaN	NaN	NaN	NaN	NaN	28
210094	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	3.0	7.0	65.0	NaN	NaN	NaN	NaN	NaN	28
210095	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	9.0	54.0	29.0	NaN	NaN	NaN	NaN	28

210096 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	statio
0	2015-10-01 01:00:00	NaN	0.8	NaN	NaN	90.0	82.0	NaN	NaN	NaN	10.0	NaN	NaN	2807900
1	2015-10-01 01:00:00	2.0	0.8	1.6	0.33	40.0	95.0	4.0	37.0	24.0	12.0	1.83	8.3	2807900
2	2015-10-01 01:00:00	3.1	NaN	1.8	NaN	29.0	97.0	NaN	NaN	NaN	NaN	NaN	7.1	2807901
3	2015-10-01 01:00:00	NaN	0.6	NaN	NaN	30.0	103.0	2.0	NaN	NaN	NaN	NaN	NaN	2807901
4	2015-10-01 01:00:00	NaN	NaN	NaN	NaN	95.0	96.0	2.0	NaN	NaN	9.0	NaN	NaN	2807901
5	2015-10-01 01:00:00	0.7	0.4	0.3	NaN	35.0	104.0	1.0	26.0	NaN	3.0	NaN	3.3	2807901
6	2015-10-01 01:00:00	0.5	0.3	0.3	0.12	6.0	83.0	1.0	19.0	12.0	3.0	1.29	4.8	2807902
7	2015-10-01 01:00:00	NaN	NaN	NaN	NaN	54.0	94.0	1.0	NaN	NaN	NaN	NaN	NaN	2807902
8	2015-10-01 01:00:00	NaN	0.5	NaN	NaN	38.0	114.0	16.0	NaN	NaN	NaN	NaN	NaN	2807903
9	2015-10-01 01:00:00	NaN	0.7	NaN	NaN	64.0	97.0	NaN	34.0	NaN	6.0	NaN	NaN	2807903

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

Out[4]:

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
210076	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	11.0	62.0	NaN	NaN	4.0	NaN	NaN	28
210077	2015-08-01 00:00:00	0.1	0.1	0.1	NaN	2.0	15.0	61.0	11.0	NaN	2.0	NaN	0.3	28
210078	2015-08-01 00:00:00	0.1	0.1	0.1	0.06	1.0	8.0	65.0	7.0	1.0	2.0	1.18	0.4	28
210079	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	7.0	63.0	NaN	NaN	NaN	NaN	NaN	28
210080	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	4.0	22.0	73.0	NaN	NaN	5.0	NaN	NaN	28
210081	2015-08-01 00:00:00	NaN	0.1	NaN	NaN	1.0	9.0	NaN	19.0	NaN	7.0	NaN	NaN	28
210082	2015-08-01 00:00:00	0.1	NaN	0.1	NaN	4.0	15.0	NaN	13.0	7.0	2.0	NaN	0.3	28
210083	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	2.0	10.0	67.0	NaN	NaN	NaN	NaN	NaN	28
210084	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	13.0	NaN	17.0	NaN	4.0	NaN	NaN	28
210085	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	17.0	NaN	12.0	6.0	NaN	NaN	NaN	28
210086	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	2.0	9.0	NaN	9.0	5.0	NaN	NaN	NaN	28
210087	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	11.0	63.0	NaN	NaN	NaN	NaN	NaN	28
210088	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	7.0	NaN	8.0	3.0	NaN	NaN	NaN	28
210089	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	2.0	14.0	64.0	NaN	NaN	NaN	NaN	NaN	28
210090	2015-08-01 00:00:00	0.2	NaN	0.2	0.48	1.0	15.0	NaN	35.0	NaN	NaN	1.59	0.2	28
210091	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	11.0	33.0	53.0	NaN	NaN	NaN	NaN	NaN	28
210092	2015-08-01 00:00:00	NaN	0.2	NaN	NaN	1.0	5.0	NaN	26.0	NaN	10.0	NaN	NaN	28

	date	BEN	CO	EBE	NMHC	NO	NO_2	O_3	PM10	PM25	SO_2	TCH	TOL	
210093	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	7.0	74.0	NaN	NaN	NaN	NaN	NaN	28
210094	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	3.0	7.0	65.0	NaN	NaN	NaN	NaN	NaN	28
210095	2015-08-01 00:00:00	NaN	NaN	NaN	NaN	1.0	9.0	54.0	29.0	NaN	NaN	NaN	NaN	28

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

Out[5]:

	BEN	CO	EBE	NMHC	NO	NO_2	
count	51039.000000	86827.000000	50962.000000	25756.000000	208805.000000	208805.000000	1
mean	0.756945	0.366559	0.495269	0.261542	26.736821	40.980302	
std	0.998673	0.285630	0.883413	0.295215	59.570283	32.731316	
min	0.100000	0.100000	0.100000	0.000000	1.000000	1.000000	
25%	0.100000	0.200000	0.100000	0.090000	2.000000	17.000000	
50%	0.400000	0.300000	0.200000	0.140000	6.000000	33.000000	
75%	1.000000	0.400000	0.500000	0.300000	22.000000	57.000000	
max	17.700001	4.500000	19.700001	2.800000	1146.000000	424.000000	

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

Out[6]:
```
(210096, 14)
```

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

Out[7]:
```
2941344
```

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
...	...	...	...	...	...	...	...	...	...	...	...	...	...
210091	False	True	False	True	True	False	False	False	True	True	True	True	True
210092	False	True	False	True	True	False	False	True	False	True	False	True	True
210093	False	True	True	True	True	False	False	False	True	True	True	True	True
210094	False	True	True	True	True	False	False	False	True	True	True	True	True
210095	False	True	True	True	True	False	False	False	False	True	True	True	True

210096 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	2015-10-01 01:00:00	2.0	0.8	1.6	0.33	40.0	95.0	4.0	37.0	24.0	12.0	1.83	8.3	28(
6	2015-10-01 01:00:00	0.5	0.3	0.3	0.12	6.0	83.0	1.0	19.0	12.0	3.0	1.29	4.8	28(
25	2015-10-01 02:00:00	1.6	0.7	1.3	0.38	81.0	105.0	4.0	36.0	19.0	13.0	1.93	6.9	28(
30	2015-10-01 02:00:00	0.4	0.3	0.3	0.11	5.0	72.0	2.0	16.0	10.0	2.0	1.27	7.8	28(
49	2015-10-01 03:00:00	2.2	0.8	1.8	0.41	111.0	104.0	4.0	35.0	20.0	14.0	2.05	13.9	28(
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
210030	2015-07-31 22:00:00	0.1	0.1	0.1	0.06	1.0	10.0	69.0	10.0	3.0	2.0	1.18	0.2	28(
210049	2015-07-31 23:00:00	0.4	0.3	0.1	0.12	3.0	28.0	56.0	15.0	7.0	12.0	1.45	1.2	28(
210054	2015-07-31 23:00:00	0.1	0.1	0.1	0.06	1.0	10.0	63.0	5.0	1.0	2.0	1.18	0.2	28(
210073	2015-08-01 00:00:00	0.1	0.3	0.1	0.11	2.0	23.0	59.0	5.0	2.0	11.0	1.44	0.6	28(
210078	2015-08-01 00:00:00	0.1	0.1	0.1	0.06	1.0	8.0	65.0	7.0	1.0	2.0	1.18	0.4	28(

16026 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']]
```

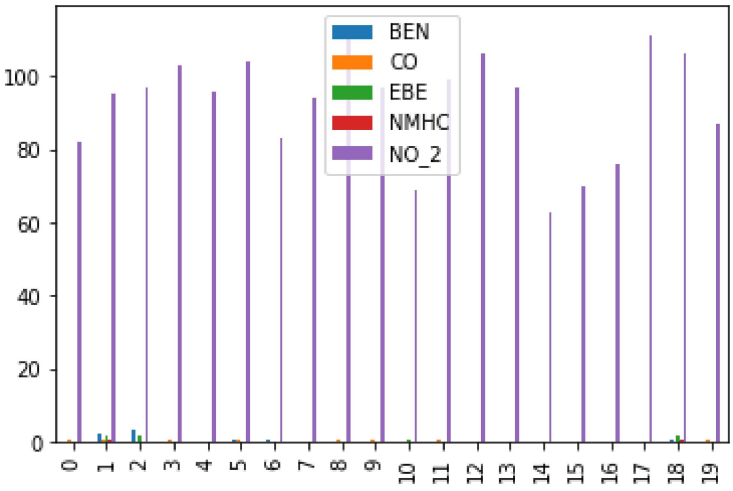
In [12]:
```
dd=sd.head(20)
dd
```

Out[12]:

	BEN	CO	EBE	NMHC	NO_2
0	NaN	0.8	NaN	NaN	82.0
1	2.0	0.8	1.6	0.33	95.0
2	3.1	NaN	1.8	NaN	97.0
3	NaN	0.6	NaN	NaN	103.0
4	NaN	NaN	NaN	NaN	96.0
5	0.7	0.4	0.3	NaN	104.0
6	0.5	0.3	0.3	0.12	83.0
7	NaN	NaN	NaN	NaN	94.0
8	NaN	0.5	NaN	NaN	114.0
9	NaN	0.7	NaN	NaN	97.0
10	0.3	NaN	0.4	NaN	69.0
11	NaN	0.6	NaN	NaN	99.0
12	NaN	NaN	NaN	NaN	106.0
13	NaN	NaN	NaN	NaN	97.0
14	NaN	NaN	NaN	NaN	63.0
15	NaN	NaN	NaN	NaN	70.0
16	NaN	NaN	NaN	NaN	76.0
17	NaN	NaN	NaN	NaN	111.0
18	0.6	NaN	1.9	0.42	106.0
19	NaN	0.7	NaN	NaN	87.0

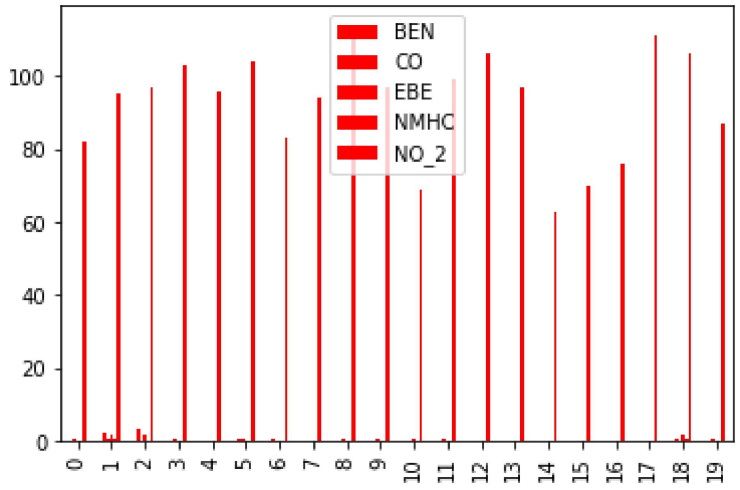
In [13]:
```
dd.plot.bar()
```

Out[13]:
```
<AxesSubplot:>
```



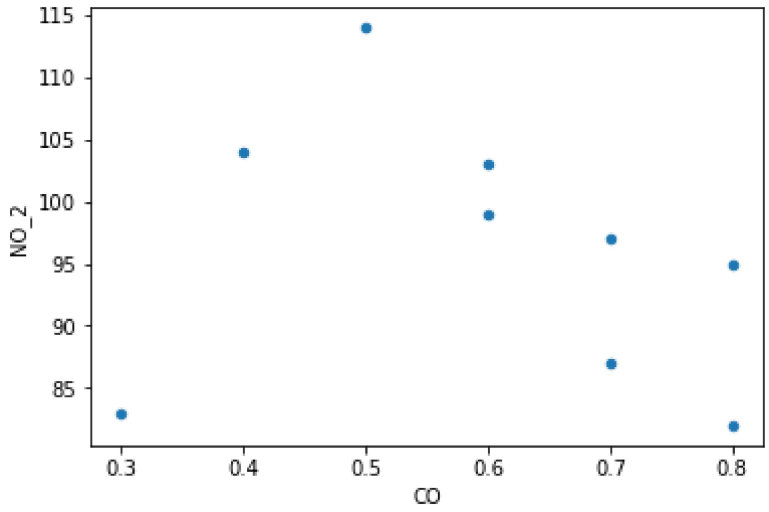
In [14]:
```
dd.plot.bar(color='r')
```

Out[14]: <AxesSubplot:>



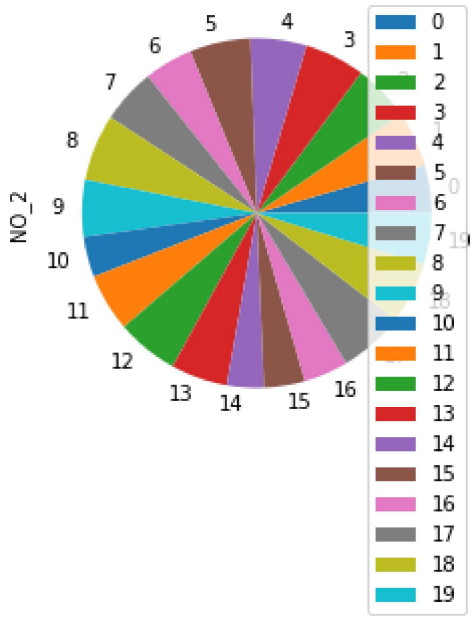
In [15]:
```
dd.plot.scatter(x='CO',y='NO_2')
```

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



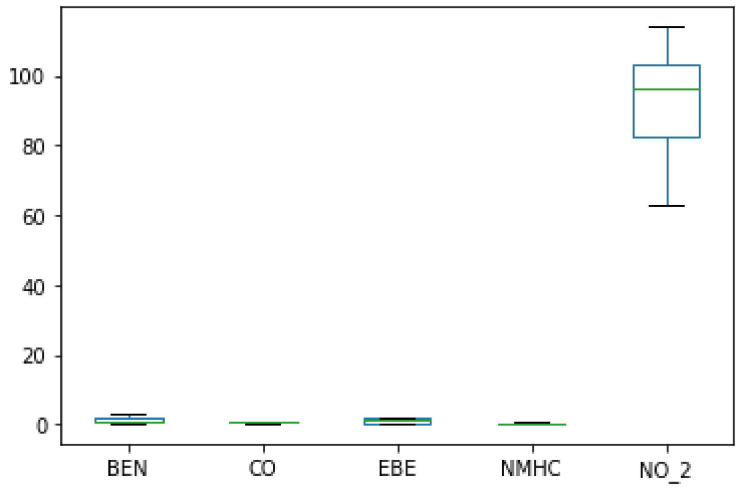
In [16]:
```
dd.plot.pie(y='NO_2')
```

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



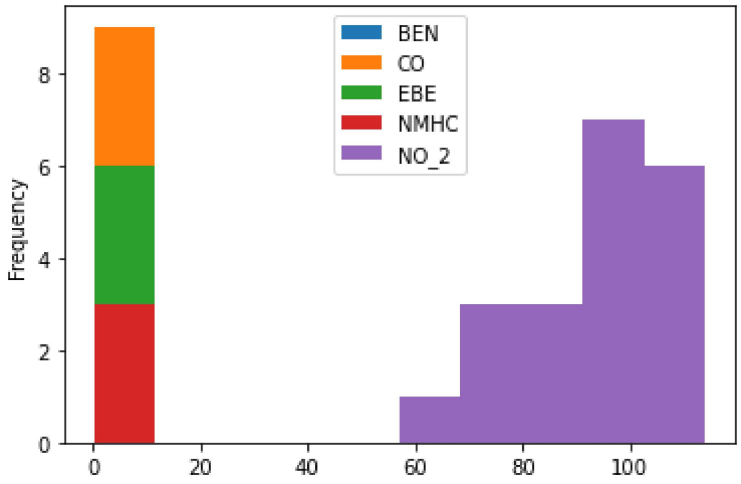
In [17]:
```
dd.plot.box()
```

Out[17]: <AxesSubplot:>



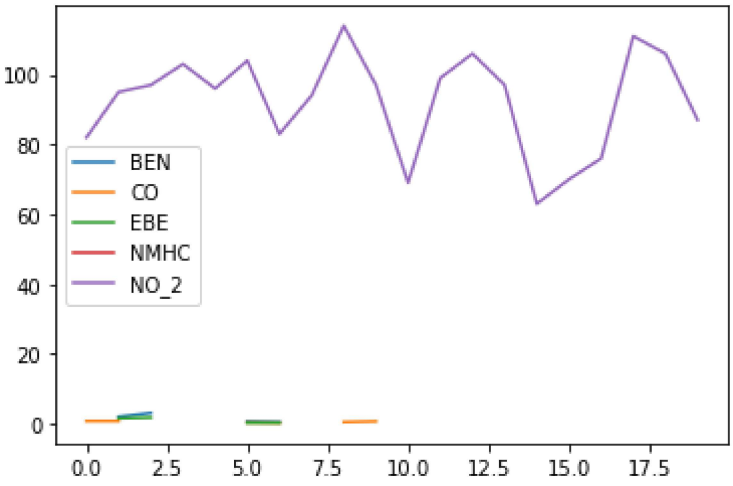
In [18]:
```
dd.plot.hist()
```

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



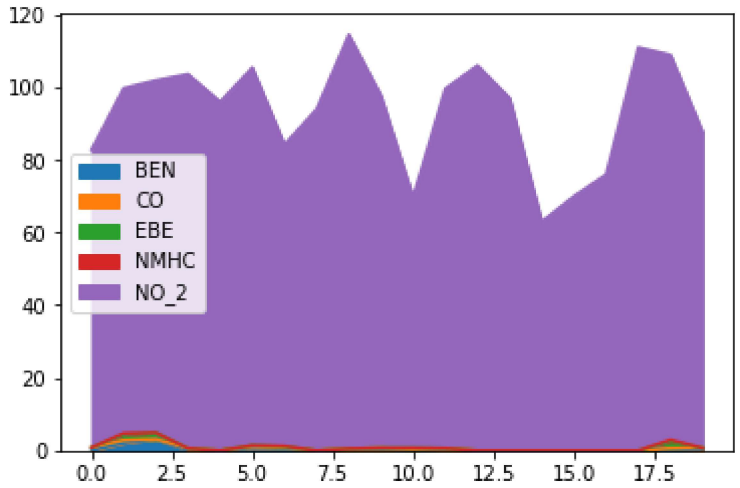
In [19]:
```
dd.plot.line()
```

Out[19]: <AxesSubplot:>



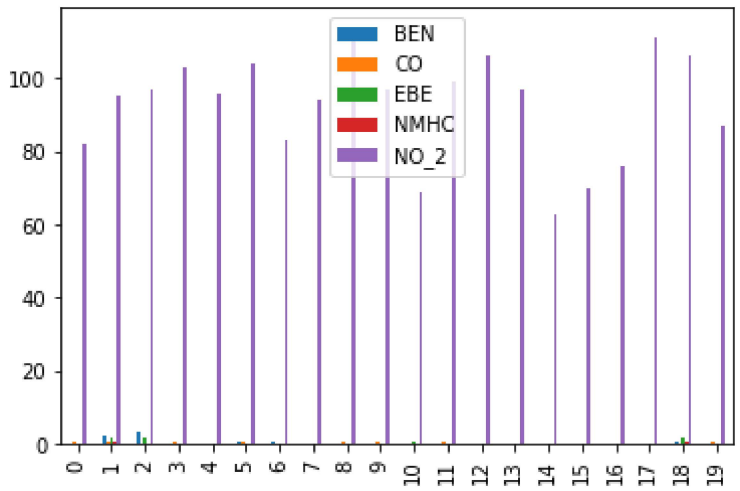
In [20]:
```
dd.plot.area()
```

Out[20]: <AxesSubplot:>



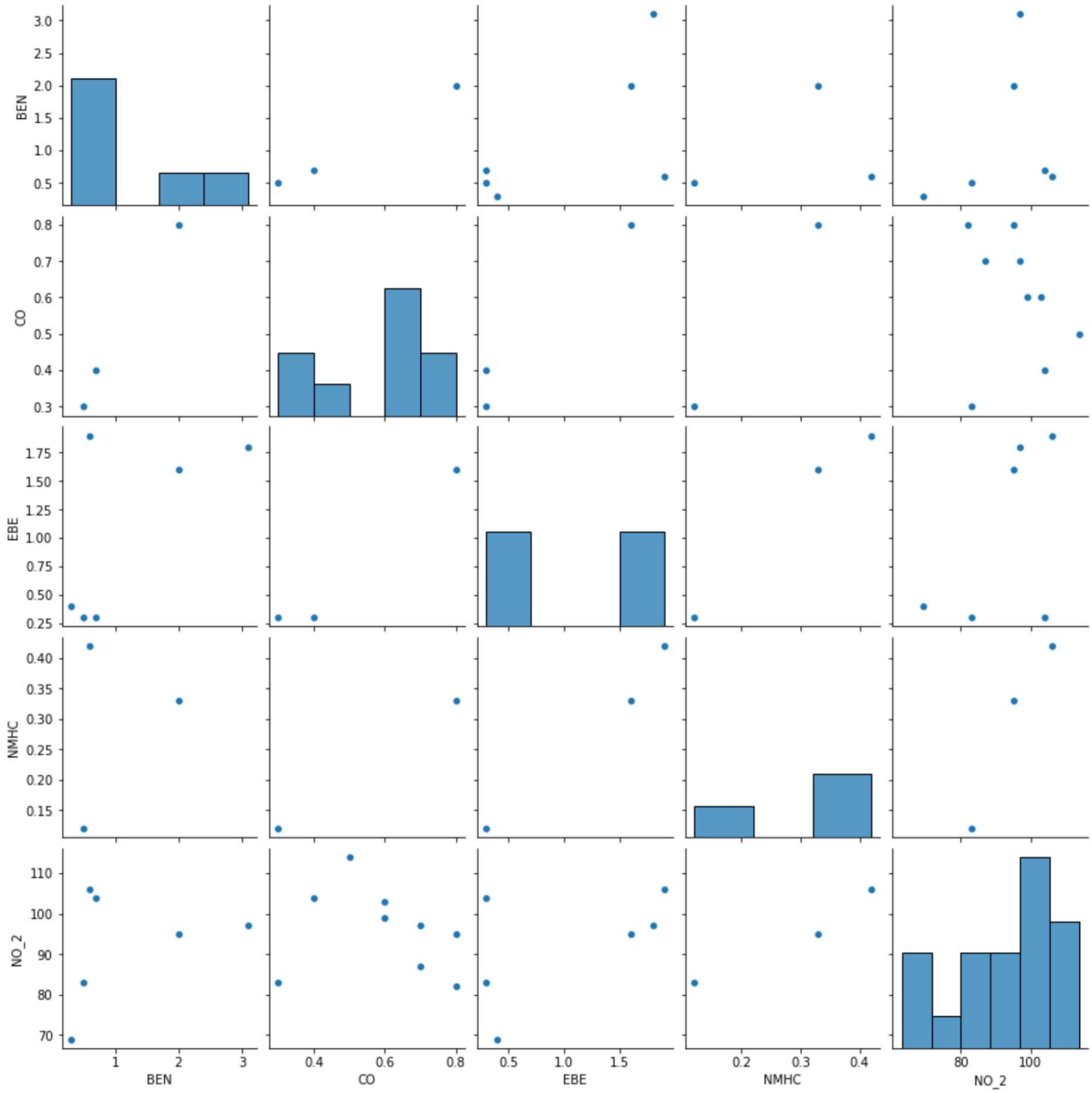
In [21]:
```
dd.plot.bar()
```

Out[21]: <AxesSubplot:>



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

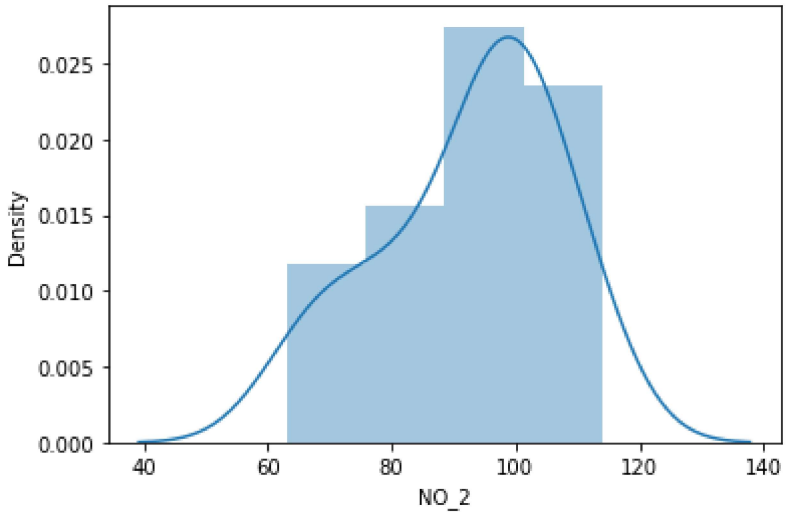
Out[22]:
```
<seaborn.axisgrid.PairGrid at 0x2312190cf70>
```



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'>



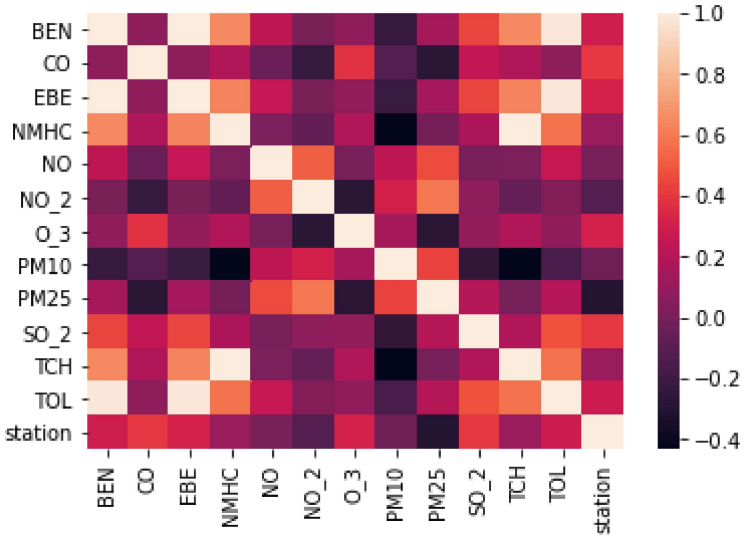
In [24]:
```
ds=data.fillna(20)
```

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_)
```

```
28079020.931648426
```

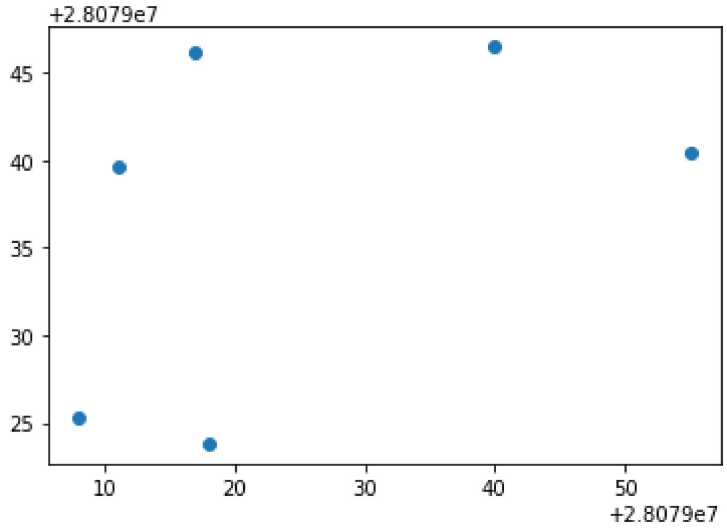
In [32]:
```
coeff= pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[32]:

	Co-efficient
BEN	0.186243
CO	0.784038
EBE	0.184561
NMHC	-0.048940
NO_2	0.032416

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

Out[33]:
```
<matplotlib.collections.PathCollection at 0x231247cabe0>
```



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

```
-0.31111095437947167
```

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

Out[35]:
```
-0.31111095437947167
```

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

Out[36]:
```
0.2843392317570176
```

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]:
```
-0.3129464656335721
```

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

Out[40]:
```
0.28430901780534046
```

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]:
```
-0.3850911218492261
```

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

Out[43]:
```
0.26857867745924624
```

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_)
```

```
[ 0.17721647  0.76530765  0.1592669  -0.          0.02902792]
```

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

```
28079021.10668583
```

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

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

```
-0.32567182260324357
```

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)
```

```
[28079054]
```

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]:
```
6.732126555297627e-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]:
```
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 [68]:
```
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[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')
```

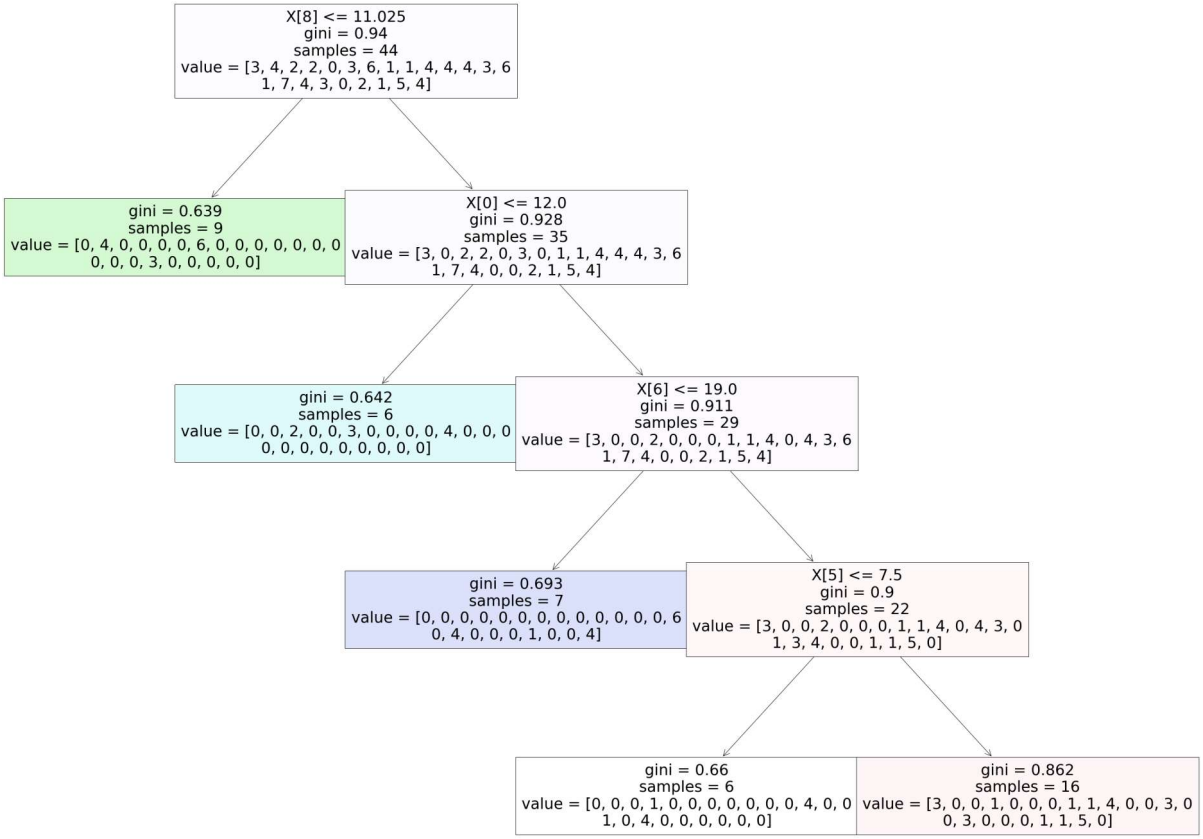
In [69]:
```
grid_search.best_score_
```

Out[69]:
```
0.4714285714285714
```

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

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

Out[71]:
```
[Text(797.1428571428571, 1956.96, 'X[8] <= 11.025\ngini = 0.94\nsamples = 44
\nvalue = [3, 4, 2, 2, 0, 3, 6, 1, 1, 4, 4, 4, 3, 6\n1, 7, 4, 3, 0, 2, 1, 5,
4]'),
 Text(398.57142857142856, 1522.0800000000002, 'gini = 0.639\nsamples = 9\nval
ue = [0, 4, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0\n0, 0, 0, 3, 0, 0, 0, 0, 0]'),
 Text(1195.7142857142858, 1522.0800000000002, 'X[0] <= 12.0\ngini = 0.928\nsa
mples = 35\nvalue = [3, 0, 2, 2, 0, 3, 0, 1, 1, 4, 4, 4, 3, 6\n1, 7, 4, 0, 0,
2, 1, 5, 4]'),
 Text(797.1428571428571, 1087.2, 'gini = 0.642\nsamples = 6\nvalue = [0, 0,
2, 0, 0, 3, 0, 0, 0, 0, 4, 0, 0, 0\n0, 0, 0, 0, 0, 0, 0, 0, 0]'),
 Text(1594.2857142857142, 1087.2, 'X[6] <= 19.0\ngini = 0.911\nsamples = 29\n
value = [3, 0, 0, 2, 0, 0, 0, 1, 1, 4, 0, 4, 3, 6\n1, 7, 4, 0, 0, 2, 1, 5,
4]'),
 Text(1195.7142857142858, 652.3200000000002, 'gini = 0.693\nsamples = 7\nvalu
e = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6\n0, 4, 0, 0, 0, 1, 0, 0, 4]'),
 Text(1992.8571428571427, 652.3200000000002, 'X[5] <= 7.5\ngini = 0.9\nsample
s = 22\nvalue = [3, 0, 0, 2, 0, 0, 0, 1, 1, 4, 0, 4, 3, 0\n1, 3, 4, 0, 0, 1,
1, 5, 0]'),
 Text(1594.2857142857142, 217.44000000000005, 'gini = 0.66\nsamples = 6\nvalu
e = [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0\n1, 0, 4, 0, 0, 0, 0, 0, 0]'),
 Text(2391.4285714285716, 217.44000000000005, 'gini = 0.862\nsamples = 16\nva
lue = [3, 0, 0, 1, 0, 0, 0, 1, 1, 4, 0, 0, 3, 0\n0, 3, 0, 0, 0, 1, 1, 5,
0]')]
```



Conclusion : Ridge,Lasso ()
0.28430901780534046 HIGH RANGE

In [ ]: