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

In [231]:
```
df=pd.read_csv(r"C:\Users\Admin\Downloads\csvs_per_year\csvs_per_year\madrid_2007.csv")
df
```

Out[231]:

| | date | BEN | CO | EBE | MXY | NMHC | NO_2 | NOx | OXY | O_3 | PM10 | PM25 | PX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2007-12-01 01:00:00 | NaN | 2.86 | NaN | NaN | NaN | 282.200012 | 1054.000000 | NaN | 4.030000 | 156.199997 | 97.43 | Na |
| **1** | 2007-12-01 01:00:00 | NaN | 1.82 | NaN | NaN | NaN | 86.419998 | 354.600006 | NaN | 3.260000 | 80.809998 | NaN | Na |
| **2** | 2007-12-01 01:00:00 | NaN | 1.47 | NaN | NaN | NaN | 94.639999 | 319.000000 | NaN | 5.310000 | 53.099998 | NaN | Na |
| **3** | 2007-12-01 01:00:00 | NaN | 1.64 | NaN | NaN | NaN | 127.900002 | 476.700012 | NaN | 4.500000 | 105.300003 | NaN | Na |
| **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 | 15.90 | 3.5 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **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 | 6.760000 | 5.14 | 1.0 |
| **225116** | 2007-03-01 00:00:00 | NaN | 0.16 | NaN | NaN | NaN | 46.820000 | 51.480000 | NaN | 22.150000 | 5.700000 | NaN | Na |
| **225117** | 2007-03-01 00:00:00 | 0.24 | NaN | 0.20 | NaN | 0.09 | 51.259998 | 66.809998 | NaN | 18.540001 | 13.010000 | 6.95 | Na |
| **225118** | 2007-03-01 00:00:00 | 0.11 | NaN | 1.00 | NaN | 0.05 | 24.240000 | 36.930000 | NaN | NaN | 6.610000 | NaN | Na |
| **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 | 10.260000 | 7.08 | 1.2 |

225120 rows × 17 columns

In [232]:
```
df1 = df.fillna(0)
df1
```

Out[232]:

| | date | BEN | CO | EBE | MXY | NMHC | NO_2 | NOx | OXY | O_3 | PM10 | PM25 | PX |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2007-12-01 01:00:00 | 0.00 | 2.86 | 0.00 | 0.00 | 0.00 | 282.200012 | 1054.000000 | 0.00 | 4.030000 | 156.199997 | 97.43 | 0.0 |
| **1** | 2007-12-01 01:00:00 | 0.00 | 1.82 | 0.00 | 0.00 | 0.00 | 86.419998 | 354.600006 | 0.00 | 3.260000 | 80.809998 | 0.00 | 0.0 |
| **2** | 2007-12-01 01:00:00 | 0.00 | 1.47 | 0.00 | 0.00 | 0.00 | 94.639999 | 319.000000 | 0.00 | 5.310000 | 53.099998 | 0.00 | 0.0 |
| **3** | 2007-12-01 01:00:00 | 0.00 | 1.64 | 0.00 | 0.00 | 0.00 | 127.900002 | 476.700012 | 0.00 | 4.500000 | 105.300003 | 0.00 | 0.0 |
| **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 | 15.90 | 3.5 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | . |
| **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 | 6.760000 | 5.14 | 1.0 |
| **225116** | 2007-03-01 00:00:00 | 0.00 | 0.16 | 0.00 | 0.00 | 0.00 | 46.820000 | 51.480000 | 0.00 | 22.150000 | 5.700000 | 0.00 | 0.0 |
| **225117** | 2007-03-01 00:00:00 | 0.24 | 0.00 | 0.20 | 0.00 | 0.09 | 51.259998 | 66.809998 | 0.00 | 18.540001 | 13.010000 | 6.95 | 0.0 |
| **225118** | 2007-03-01 00:00:00 | 0.11 | 0.00 | 1.00 | 0.00 | 0.05 | 24.240000 | 36.930000 | 0.00 | 0.000000 | 6.610000 | 0.00 | 0.0 |
| **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 | 10.260000 | 7.08 | 1.2 |

225120 rows × 17 columns

In [233]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 225120 entries, 0 to 225119
Data columns (total 17 columns):
 #   Column   Non-Null Count   Dtype  
---  ------   --------------   -----  
 0   date     225120 non-null  object 
 1   BEN      68885 non-null   float64
 2   CO       206748 non-null  float64
 3   EBE      68883 non-null   float64
 4   MXY      26061 non-null   float64
 5   NMHC     86883 non-null   float64
 6   NO_2     223985 non-null  float64
 7   NOx      223972 non-null  float64
 8   OXY      26062 non-null   float64
 9   O_3      211850 non-null  float64
 10  PM10     222588 non-null  float64
 11  PM25     68870 non-null   float64
 12  PXY      26062 non-null   float64
 13  SO_2     224372 non-null  float64
 14  TCH      87026 non-null   float64
 15  TOL      68845 non-null   float64
 16  station  225120 non-null  int64  
dtypes: float64(15), int64(1), object(1)
memory usage: 29.2+ MB
```

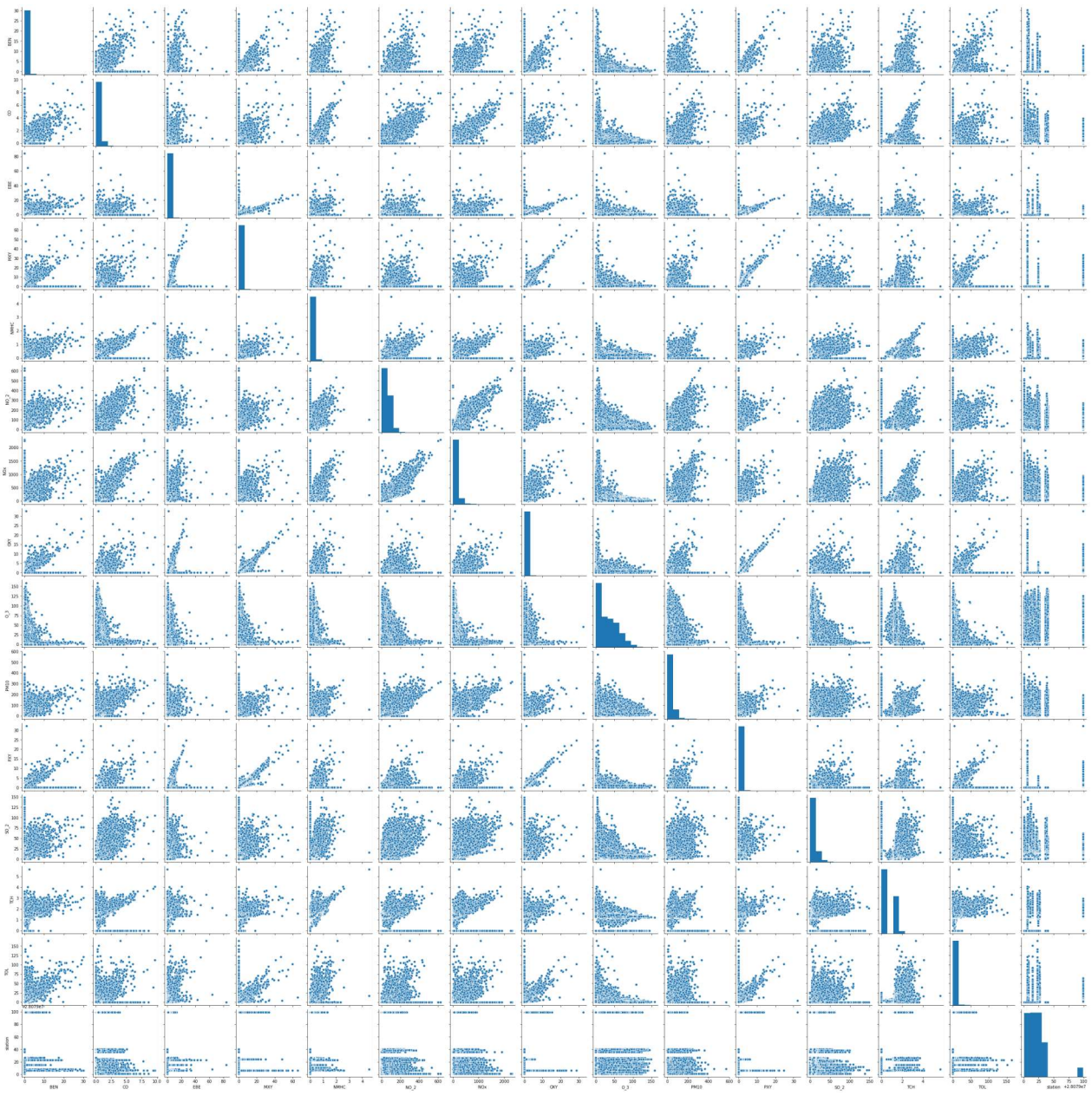
In [234]:
```
df.columns
```

Out[234]:
```
Index(['date', 'BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
       'PM10', 'PM25', 'PXY', 'SO_2', 'TCH', 'TOL', 'station'],
      dtype='object')
```

In [235]:
```
df2=df1[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY', 'O_3',
       'PM10', 'PXY', 'SO_2', 'TCH', 'TOL', 'station']]
```

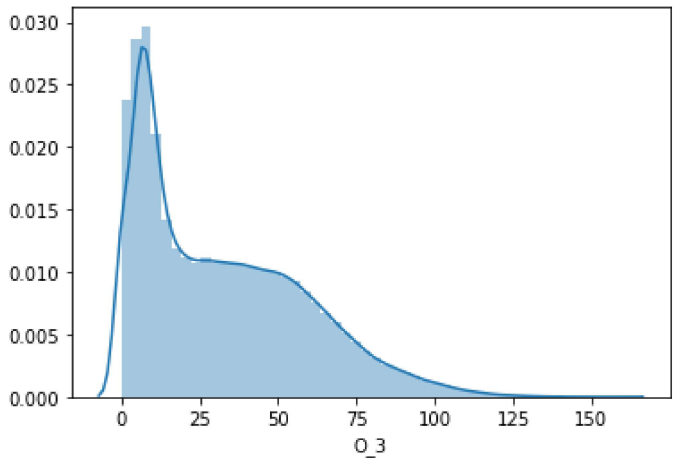
In [236]:
```
sns.pairplot(df2)
```

Out[236]: <seaborn.axisgrid.PairGrid at 0x2347e8248b0>



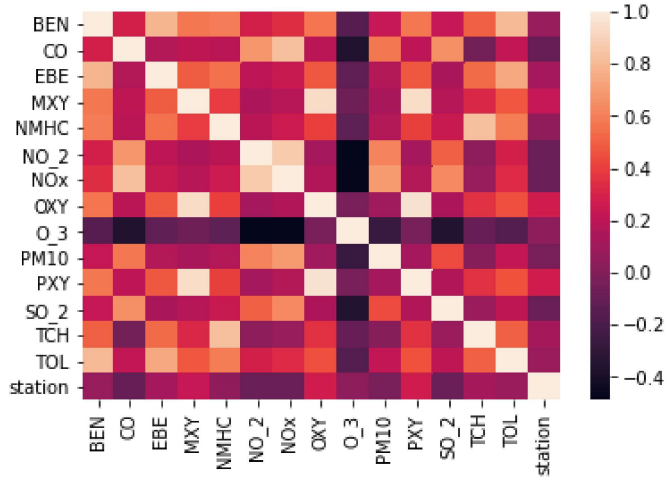
In [237]:
```
sns.distplot(df2['O_3'])
```

Out[237]: <matplotlib.axes._subplots.AxesSubplot at 0x234c6be6100>




In [238]:
```
sns.heatmap(df2.corr())
```

Out[238]: <matplotlib.axes._subplots.AxesSubplot at 0x234c6be6d90>




# Linear Regression

In [239]:
```
x = df2[['BEN', 'CO', 'EBE', 'MXY', 'NMHC', 'NO_2', 'NOx', 'OXY',
       'PM10', 'PXY', 'SO_2', 'TCH', 'TOL']]
y = df2['O_3']
```

In [240]:
```
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 [241]:
```
from sklearn.linear_model import LinearRegression

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

Out[241]: LinearRegression()

In [242]:
```
print(lr.intercept_)
```

```
53.15688488180901
```

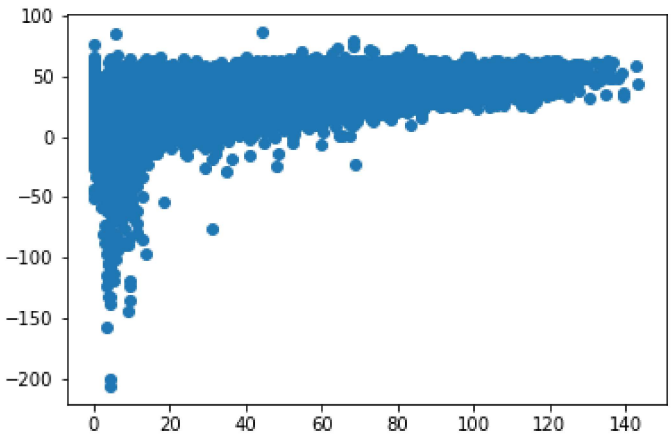
In [243]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-effecient'])
coeff
```

Out[243]:

| | Co-effecient |
|---|---|
| **BEN** | -0.057562 |
| **CO** | 2.077285 |
| **EBE** | 0.719431 |
| **MXY** | -5.469133 |
| **NMHC** | 31.356843 |
| **NO_2** | -0.215298 |
| **NOx** | -0.062534 |
| **OXY** | 5.615105 |
| **PM10** | 0.111524 |
| **PXY** | 8.962120 |
| **SO_2** | -0.384667 |
| **TCH** | -8.494754 |
| **TOL** | -0.117776 |

In [244]:
```
prediction=lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[244]:
```
<matplotlib.collections.PathCollection at 0x233e3730730>
```




In [245]:
```
print(lr.score(x_test,y_test))
```

```
0.28476694103229183
```

In [246]:
```
lr.score(x_train,y_train)
```

Out[246]:
```
0.28977068052875854
```

# Ridge Lasso

In [247]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [248]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

Out[248]: 0.2847685865505669

In [249]:
```
predict2=(rr.predict(x_test))
```

In [250]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[250]: Lasso(alpha=10)

In [251]:
```
la.score(x_test,y_test)
```

Out[251]: 0.25790771735023466

# Elastic Net regression

In [252]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[252]: ElasticNet()

In [253]:
```
print(en.coef_)
```

```
[ 0.          0.          0.          0.03706599  0.         -0.22328694
 -0.05210272  0.19875639  0.11845858  0.0607027  -0.31606343 -0.7221122
 -0.        ]
```

In [254]:
```
print(en.intercept_)
```

```
51.473671373914826
```

In [255]:
```
print(en.score(x_test,y_test))
```

```
0.26486212600386894
```

In [256]:
```
print(en.score(x_train,y_train))
```

```
0.2696993157810026
```

# Logistic Regression

In [257]:
```
from sklearn.linear_model import LogisticRegression
```

In [258]:
```
feature_matrix=df2.iloc[:,0:5]
target_vector=df2.iloc[:,-1]
```

In [259]:
```
from sklearn.preprocessing import StandardScaler
```

In [260]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [261]:
```
logr=LogisticRegression()
logr.fit(fs,target_vector)
```

```
C:\Users\Admin\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:762: Convergen
ceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/st
able/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://
scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[261]:
```
LogisticRegression()
```

In [262]:
```
df2.shape
```

Out[262]:
```
(225120, 15)
```

In [263]:
```
observation=[[1,2,3,4,5]]
predication = logr.predict(observation)
```

In [264]:
```
print(predication)
```

```
[28079024]
```

In [265]:
```
logr.classes_
```

Out[265]:
```
array([28079001, 28079003, 28079004, 28079006, 28079007, 28079008,
       28079009, 28079011, 28079012, 28079014, 28079015, 28079016,
       28079018, 28079019, 28079021, 28079022, 28079023, 28079024,
       28079025, 28079026, 28079027, 28079036, 28079038, 28079039,
       28079040, 28079099], dtype=int64)
```

In [266]:
```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(feature_matrix,target_vector,test_size=0.30
```

In [267]:
```
print(logr.score(x_test,y_test))
```

```
0.06951847903340441
```

In [268]:
```
print(logr.score(x_train,y_train))
```

```
0.0694106000609199
```

# Conclusion

Linear Regression is bestfit model

The Score x_test,y_test is 0.28476694103229183 and x_train,y_train score is 0.28977068052875854

In [ ]:

In [ ]: