Application of MI in Industries

Experiment 4

Dhruv Singhal || 500075346 || R177219074 || AIML || B3 || Sem 5

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot  as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
#Reading data
data=pd.read_csv("AirQualityUCI.csv")

print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9357 entries, 0 to 9356
Data columns (total 15 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Date           9357 non-null   object 
 1   Time           9357 non-null   object 
 2   CO(GT)         9357 non-null   float64
 3   PT08.S1(CO)    9357 non-null   int64  
 4   NMHC(GT)       9357 non-null   int64  
 5   C6H6(GT)       9357 non-null   float64
 6   PT08.S2(NMHC)  9357 non-null   int64  
 7   NOx(GT)        9357 non-null   int64  
 8   PT08.S3(NOx)   9357 non-null   int64  
 9   NO2(GT)        9357 non-null   int64  
 10  PT08.S4(NO2)   9357 non-null   int64  
 11  PT08.S5(O3)    9357 non-null   int64  
 12  T              9357 non-null   float64
 13  RH             9357 non-null   float64
```

```
 14  AH              9357 non-null   float64
dtypes: float64(5), int64(8), object(2)
memory usage: 1.1+ MB
None
```

In [4]:

```
print(data.describe())
```

```
            CO(GT)  PT08.S1(CO)     NMHC(GT)     C6H6(GT)  PT08.S2(NMHC)  \
count  9357.000000  9357.000000  9357.000000  9357.000000    9357.000000   
mean    -34.207524  1048.990061  -159.090093     1.865683     894.595276   
std      77.657170   329.832710   139.789093    41.380206     342.333252   
min    -200.000000  -200.000000  -200.000000  -200.000000    -200.000000   
25%       0.600000   921.000000  -200.000000     4.000000     711.000000   
50%       1.500000  1053.000000  -200.000000     7.900000     895.000000   
75%       2.600000  1221.000000  -200.000000    13.600000    1105.000000   
max      11.900000  2040.000000  1189.000000    63.700000    2214.000000   

           NOx(GT)  PT08.S3(NOx)      NO2(GT)  PT08.S4(NO2)  PT08.S5(O3)  \
count  9357.000000   9357.000000  9357.000000   9357.000000  9357.000000   
mean    168.616971    794.990168    58.148873   1391.479641   975.072032   
std     257.433866    321.993552   126.940455    467.210125   456.938184   
min    -200.000000   -200.000000  -200.000000   -200.000000  -200.000000   
25%      50.000000    637.000000    53.000000   1185.000000   700.000000   
50%     141.000000    794.000000    96.000000   1446.000000   942.000000   
75%     284.000000    960.000000   133.000000   1662.000000  1255.000000   
max    1479.000000   2683.000000   340.000000   2775.000000  2523.000000   

                 T           RH           AH  
count  9357.000000  9357.000000  9357.000000  
mean      9.778305    39.485380    -6.837604  
std      43.203623    51.216145    38.976670  
min    -200.000000  -200.000000  -200.000000  
25%      10.900000    34.100000     0.692300  
50%      17.200000    48.600000     0.976800  
75%      24.100000    61.900000     1.296200  
max      44.600000    88.700000     2.231000  
```

In [5]:

```
print(data.isnull().sum())
```

```
Date           0
Time           0
CO(GT)         0
PT08.S1(CO)    0
```

```
NMHC(GT)        0
C6H6(GT)        0
PT08.S2(NMHC)   0
NOx(GT)         0
PT08.S3(NOx)    0
NO2(GT)         0
PT08.S4(NO2)    0
PT08.S5(O3)     0
T               0
RH              0
AH              0
dtype: int64
```

In [6]:

```
print(data.corr())
```

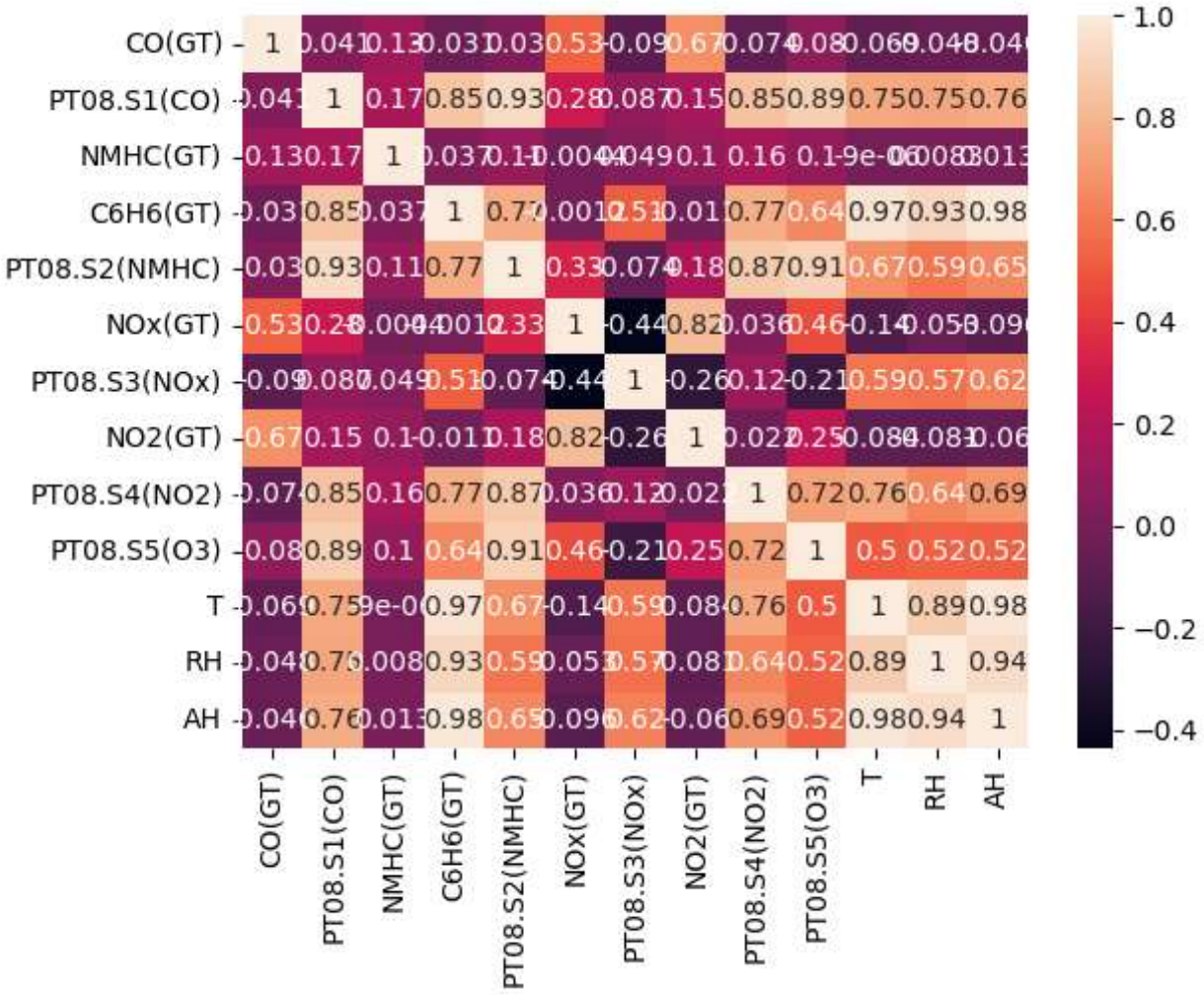
```
                CO(GT)  PT08.S1(CO)  NMHC(GT)  C6H6(GT)  PT08.S2(NMHC)  \
CO(GT)        1.000000     0.041411  0.128351 -0.031378       0.029926   
PT08.S1(CO)   0.041411     1.000000  0.170007  0.852687       0.933102   
NMHC(GT)      0.128351     0.170007  1.000000  0.037323       0.110104   
C6H6(GT)     -0.031378     0.852687  0.037323  1.000000       0.767433   
PT08.S2(NMHC) 0.029926     0.933102  0.110104  0.767433       1.000000   
NOx(GT)       0.526451     0.277993 -0.004427 -0.001174       0.331272   
PT08.S3(NOx) -0.089981     0.087019  0.048821  0.512193      -0.073667   
NO2(GT)       0.671127     0.154030  0.103307 -0.010992       0.176488   
PT08.S4(NO2) -0.073724     0.845149  0.162680  0.774673       0.874782   
PT08.S5(O3)   0.080310     0.892434  0.101185  0.641334       0.909905   
T            -0.068939     0.754844 -0.000009  0.971375       0.669025   
RH           -0.048227     0.745375  0.008284  0.925062       0.585803   
AH           -0.045892     0.764903  0.012500  0.984555       0.646572   

                NOx(GT)  PT08.S3(NOx)   NO2(GT)  PT08.S4(NO2)  PT08.S5(O3)  \
CO(GT)         0.526451     -0.089981  0.671127     -0.073724     0.080310   
PT08.S1(CO)    0.277993      0.087019  0.154030      0.845149     0.892434   
NMHC(GT)      -0.004427      0.048821  0.103307      0.162680     0.101185   
C6H6(GT)      -0.001174      0.512193 -0.010992      0.774673     0.641334   
PT08.S2(NMHC)  0.331272     -0.073667  0.176488      0.874782     0.909905   
NOx(GT)        1.000000     -0.436084  0.817139      0.035546     0.461889   
PT08.S3(NOx)  -0.436084      1.000000 -0.256232      0.122734    -0.208865   
NO2(GT)        0.817139     -0.256232  1.000000     -0.022174     0.253439   
PT08.S4(NO2)   0.035546      0.122734 -0.022174      1.000000     0.723690   
PT08.S5(O3)    0.461889     -0.208865  0.253439      0.723690     1.000000   
T             -0.138452      0.588111 -0.084104      0.755060     0.503700   
RH            -0.053009      0.573549 -0.081305      0.640707     0.524955   
AH            -0.095847      0.621618 -0.060440      0.691913     0.519467   
```

```
                      T        RH        AH
CO(GT)        -0.068939 -0.048227 -0.045892
PT08.S1(CO)    0.754844  0.745375  0.764903
NMHC(GT)      -0.000009  0.008284  0.012500
C6H6(GT)       0.971375  0.925062  0.984555
PT08.S2(NMHC)  0.669025  0.585803  0.646572
NOx(GT)       -0.138452 -0.053009 -0.095847
PT08.S3(NOx)   0.588111  0.573549  0.621618
NO2(GT)       -0.084104 -0.081305 -0.060440
PT08.S4(NO2)   0.755060  0.640707  0.691913
PT08.S5(O3)    0.503700  0.524955  0.519467
T              1.000000  0.885911  0.981001
RH             0.885911  1.000000  0.943995
AH             0.981001  0.943995  1.000000
```

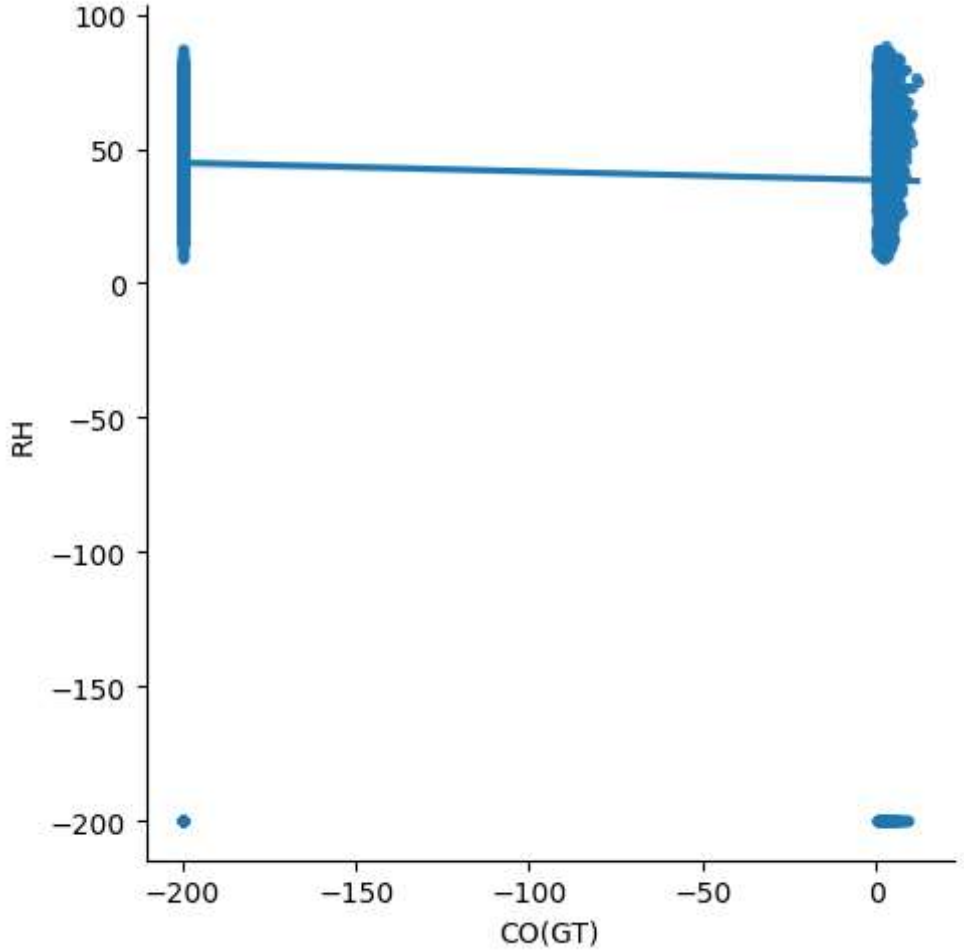
In [7]:

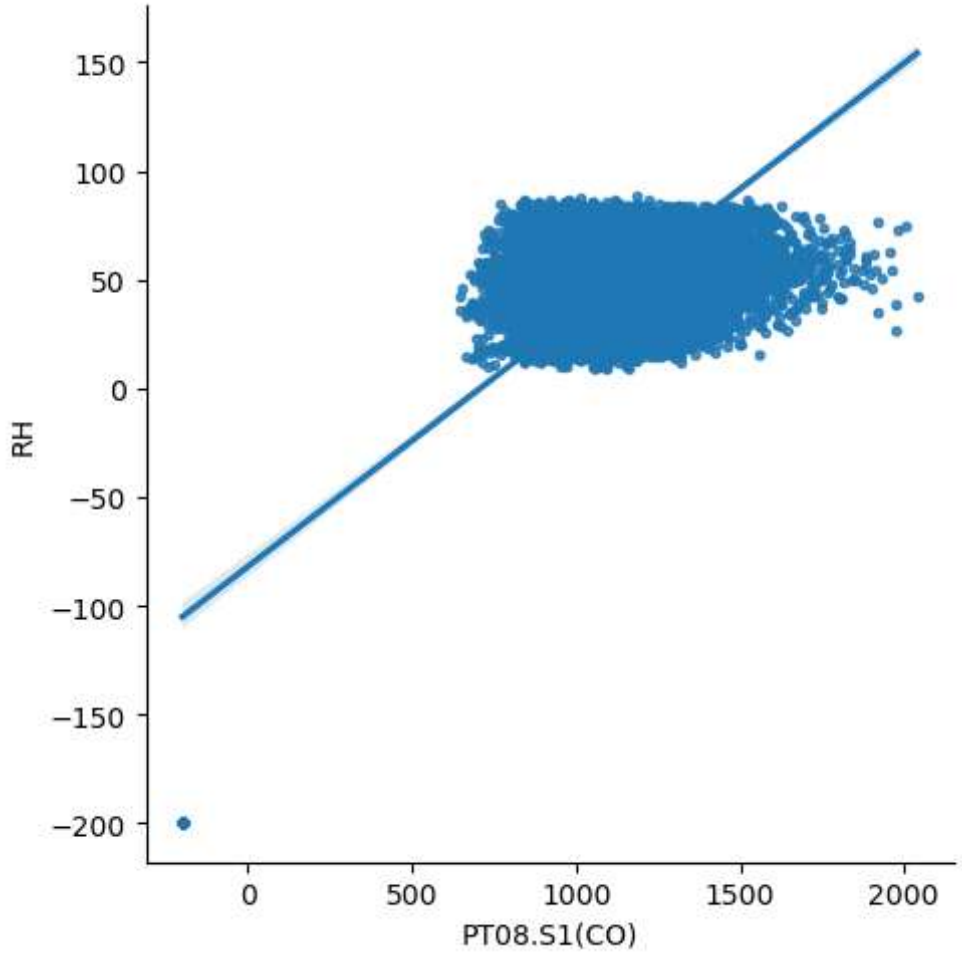
```
sns.heatmap(data.corr(), annot=True)
plt.show()
```

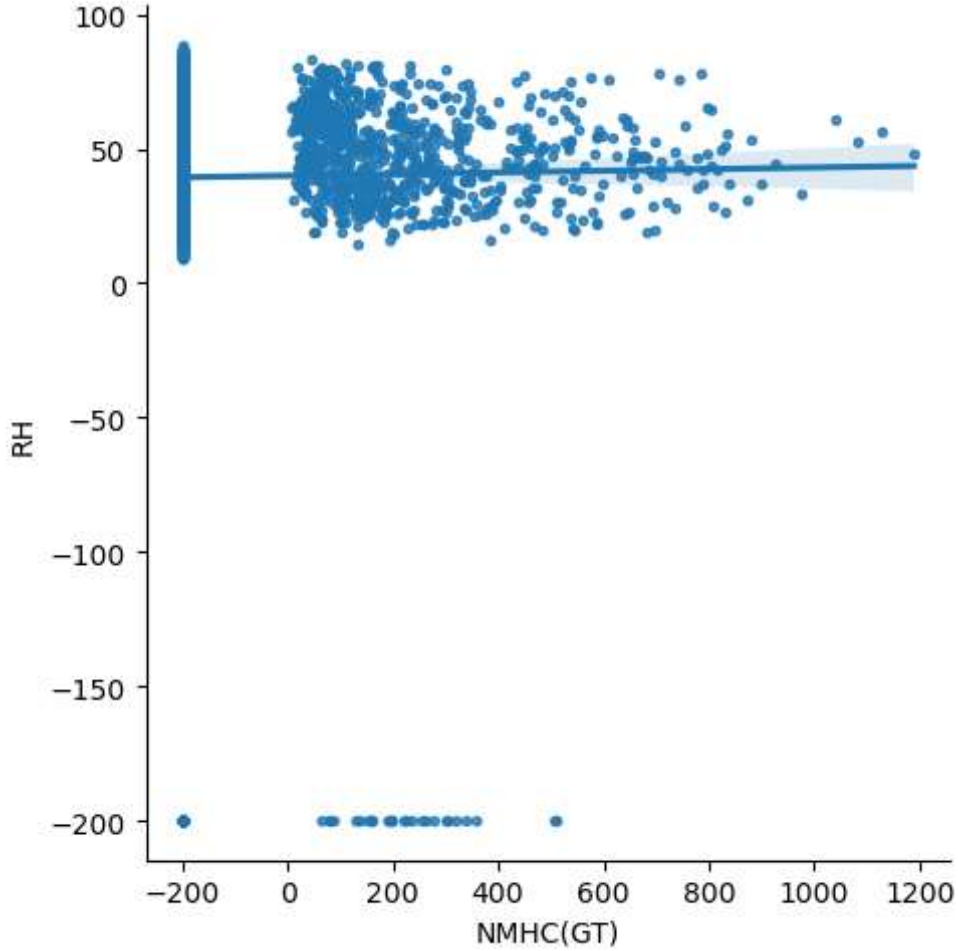


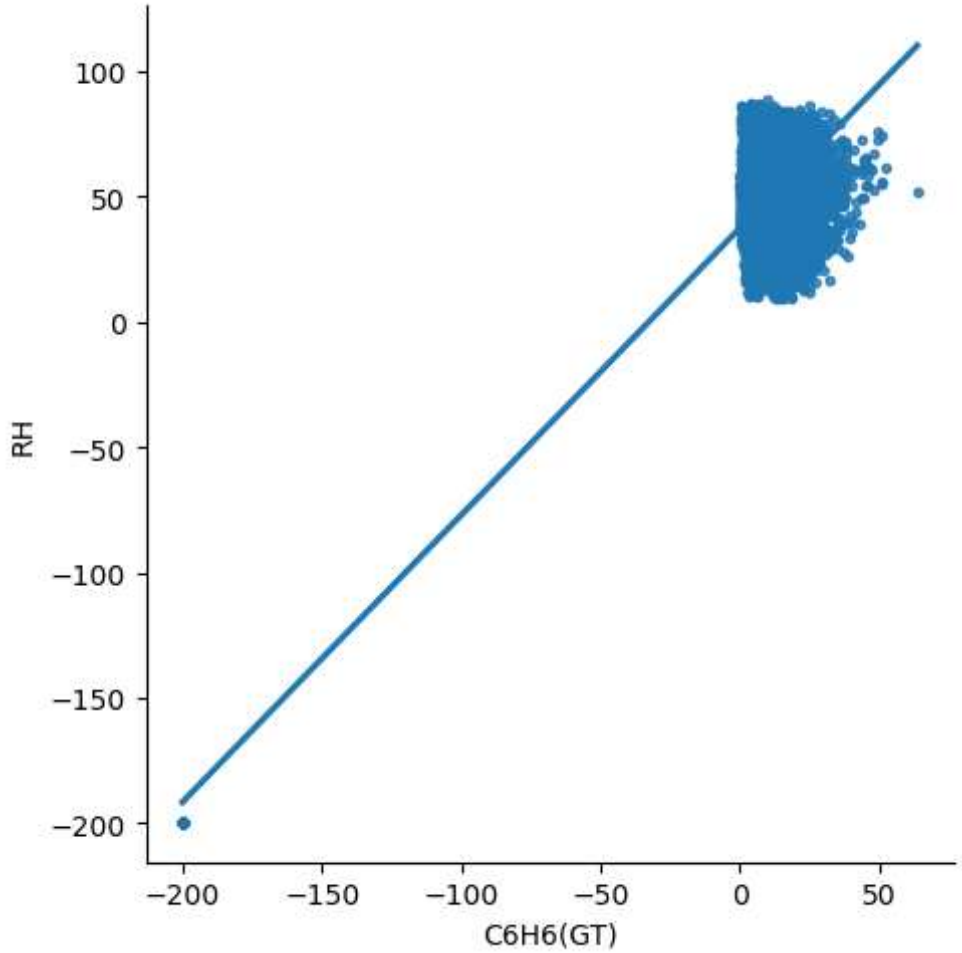
In [8]:

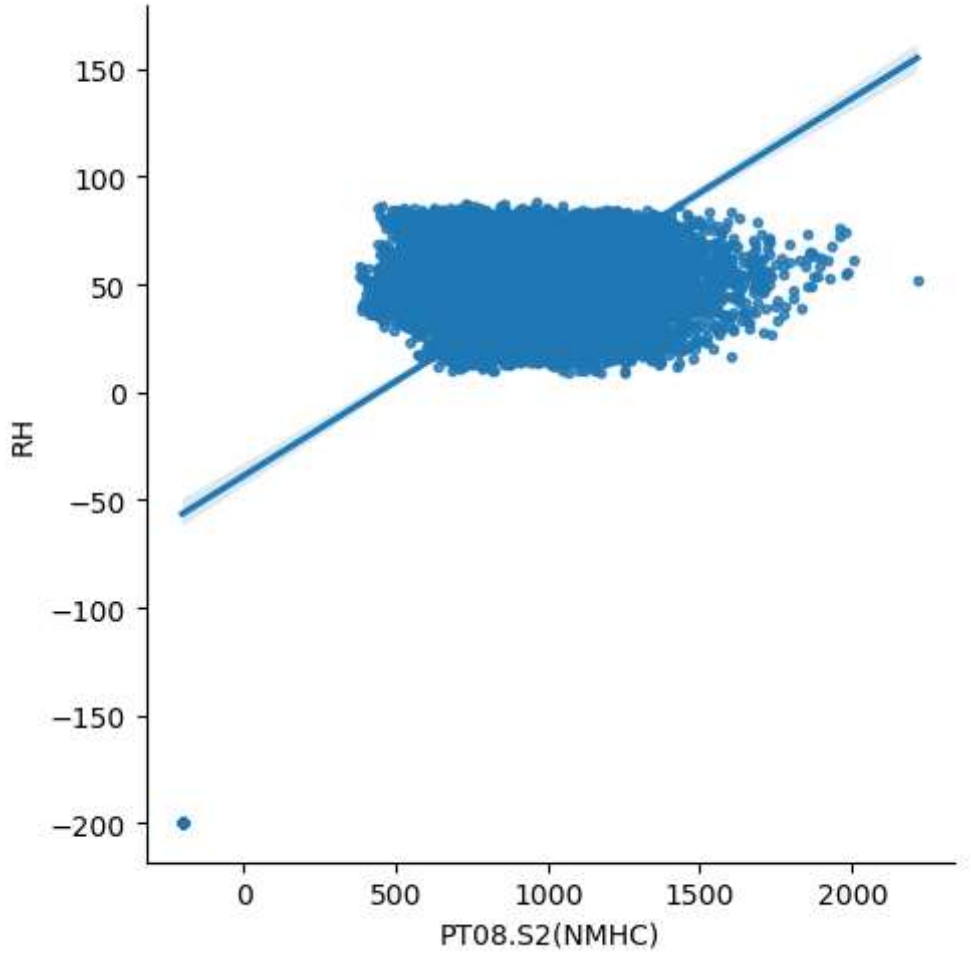
```
col_=data.columns.tolist()[2:]
for i in data.columns.tolist()[2:]:
    sns.lmplot(x=i,y='RH',data=data,markers='.')
    plt.show()
```

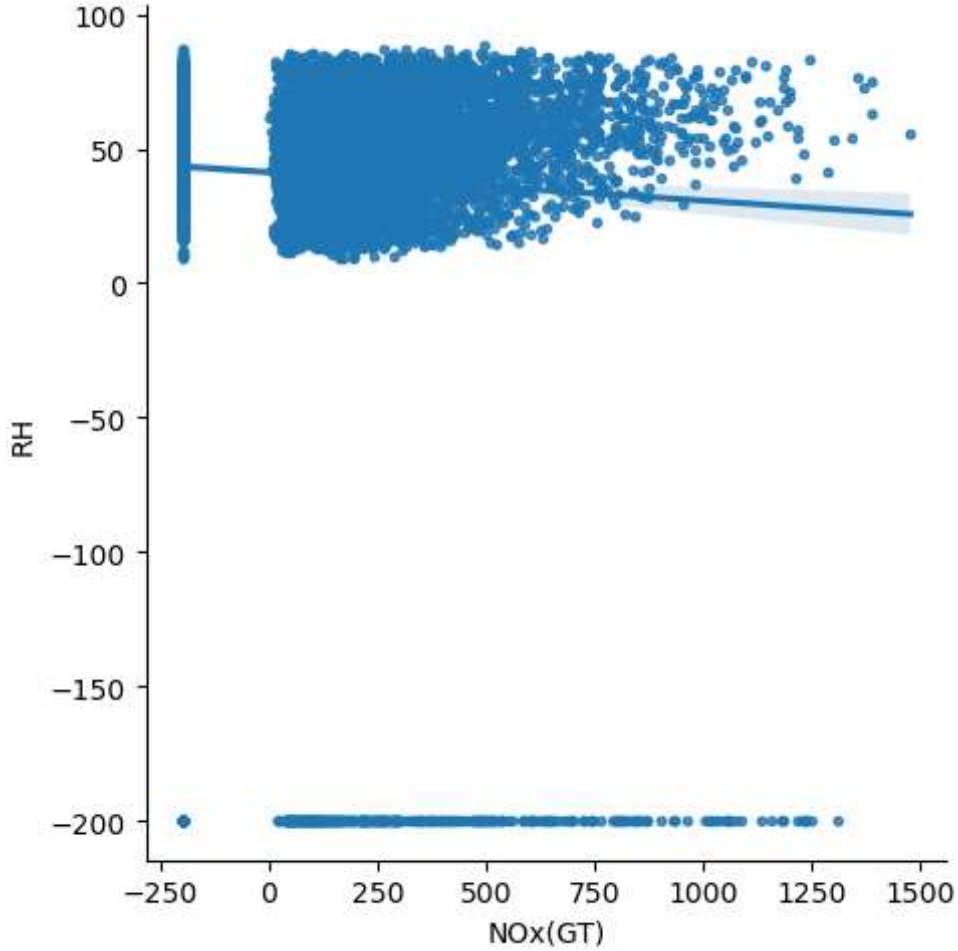

100
50
0
−50
−100
−150
−200
RH
−200
−150
−100
−50
0
CO(GT)

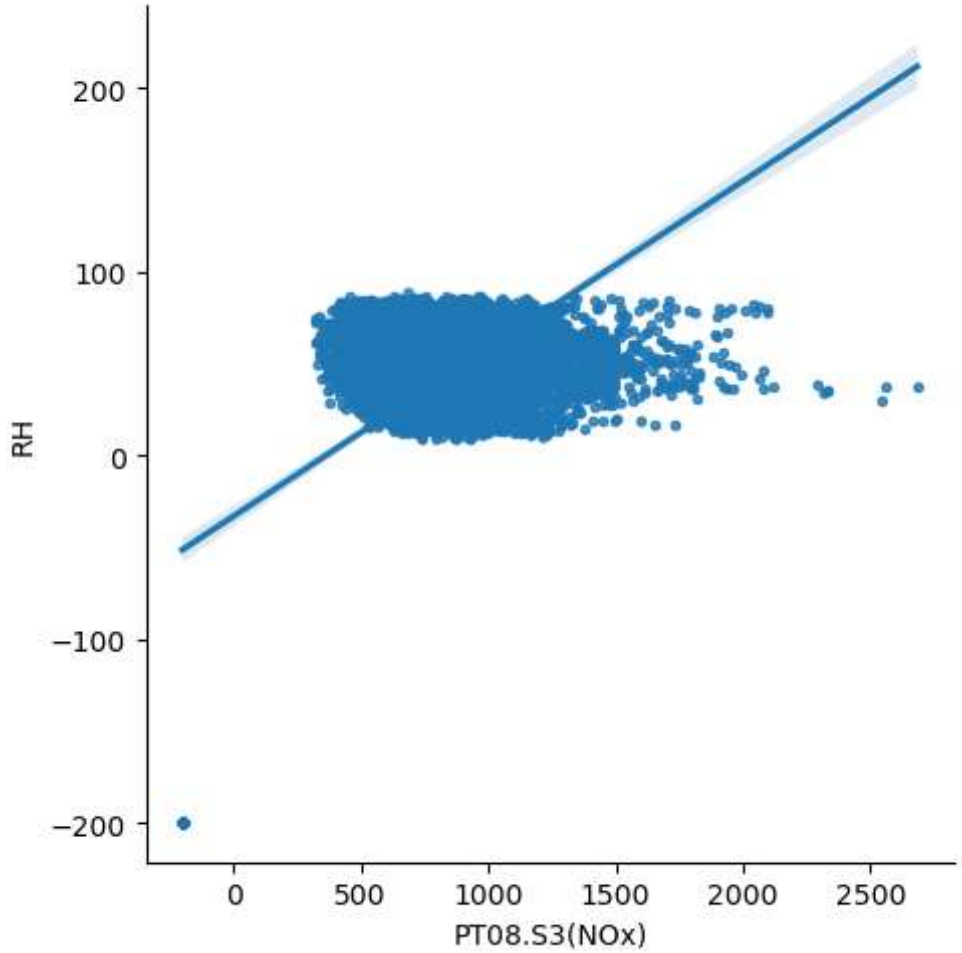

150
100
50
0
−50
−100
−150
−200
RH
0
500
1000
1500
2000
PT08.S1(CO)

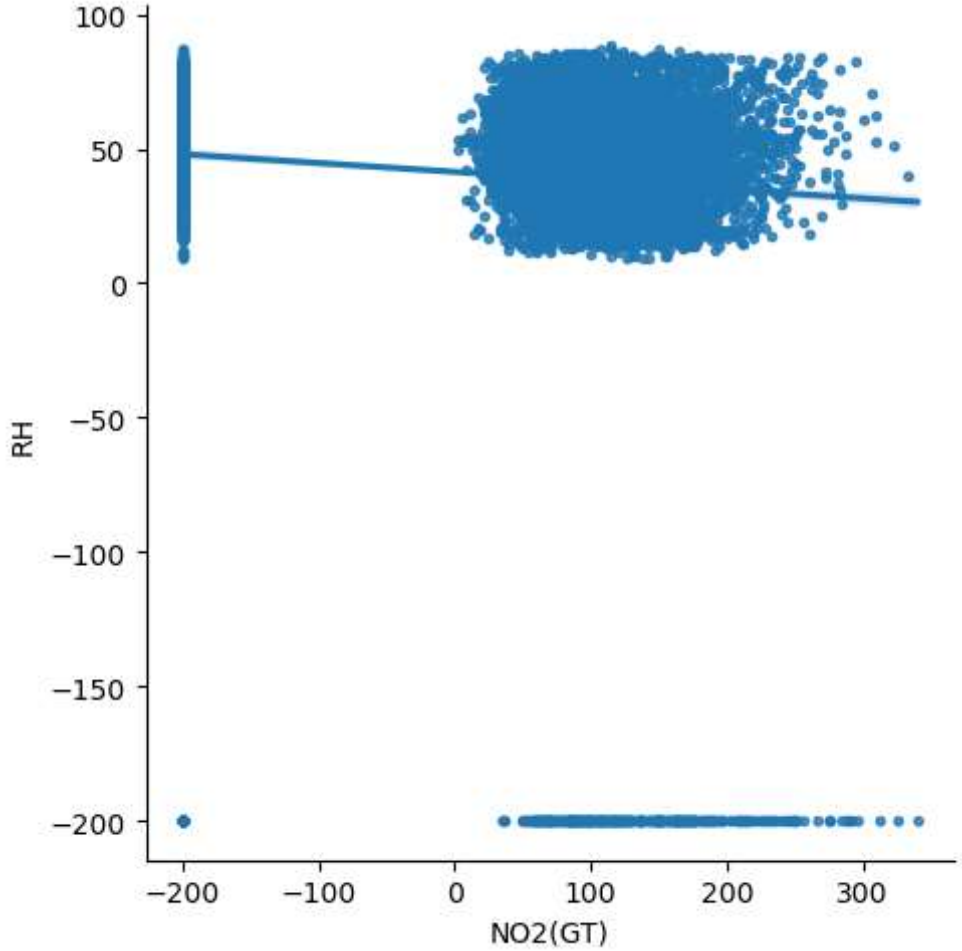

100
50
0
−50
−100
−150
−200
RH
−200
0
200
400
600
800
1000
1200
NMHC(GT)

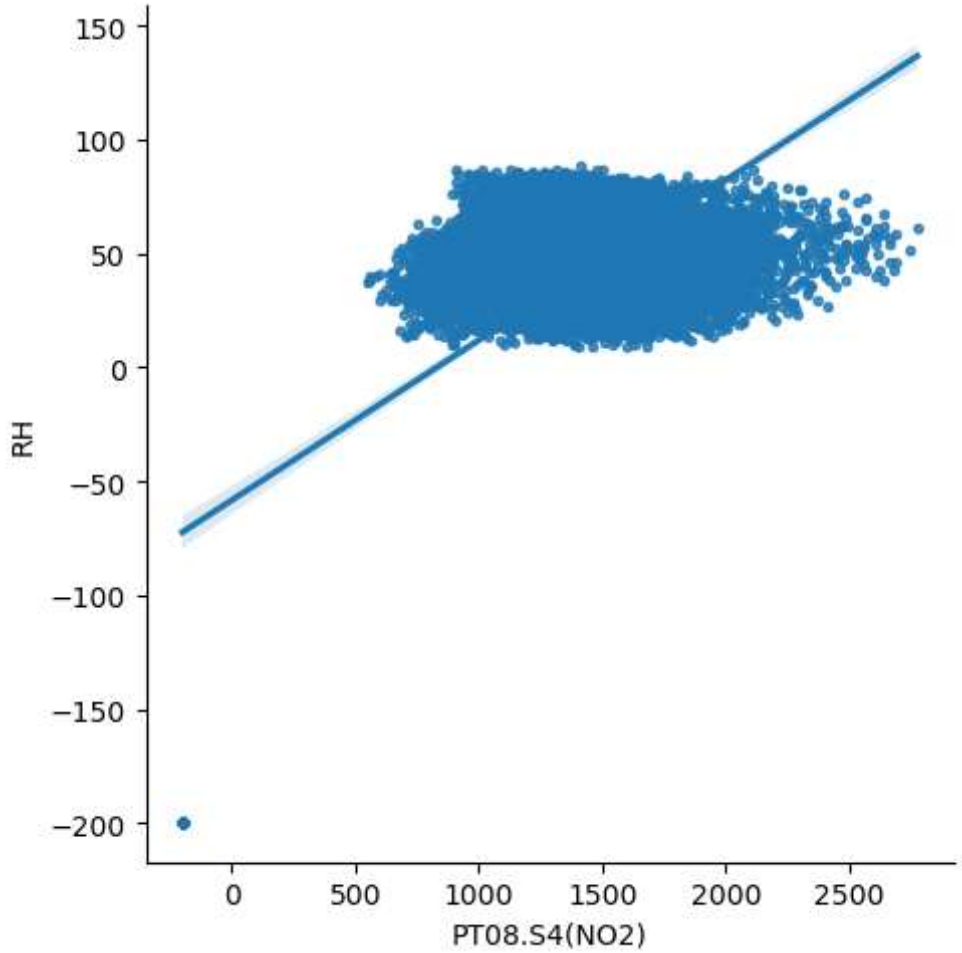

100
50
0
−50
−100
−150
−200
RH
−200
−150
−100
−50
0
50
C6H6(GT)

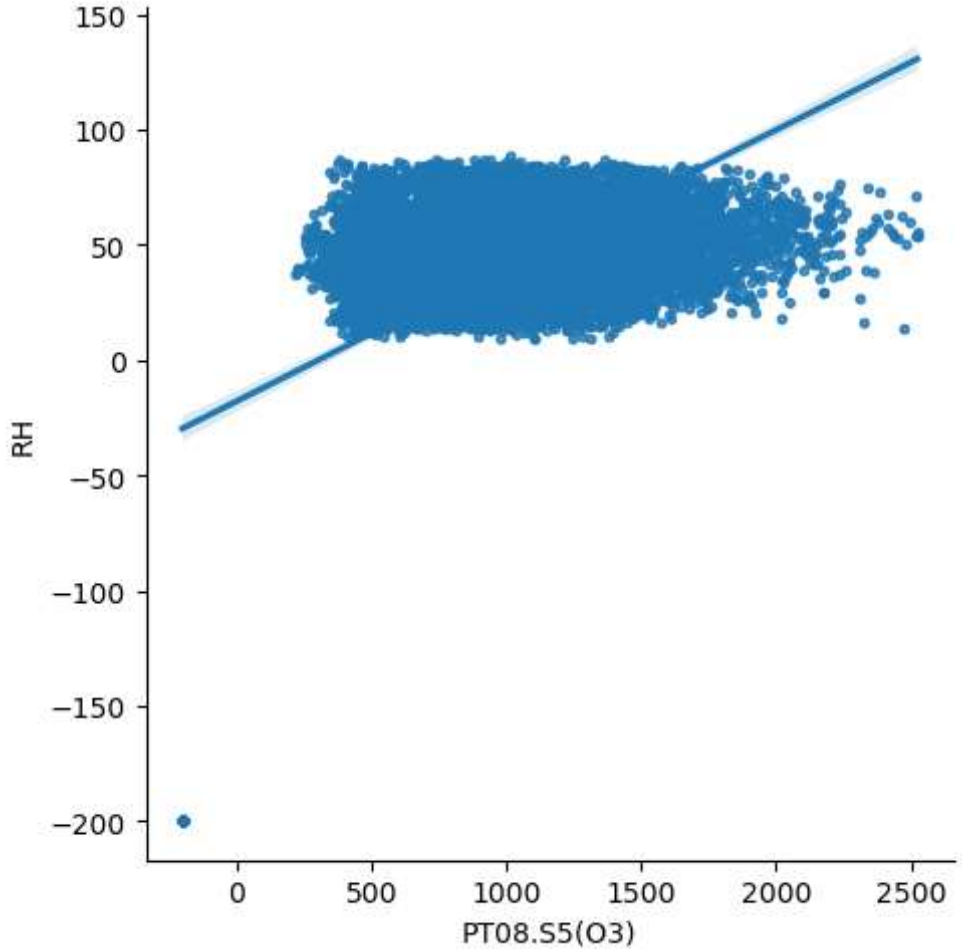

150
100
50
0
−50
−100
−150
−200
RH
0
500
1000
1500
2000
PT08.S2(NMHC)

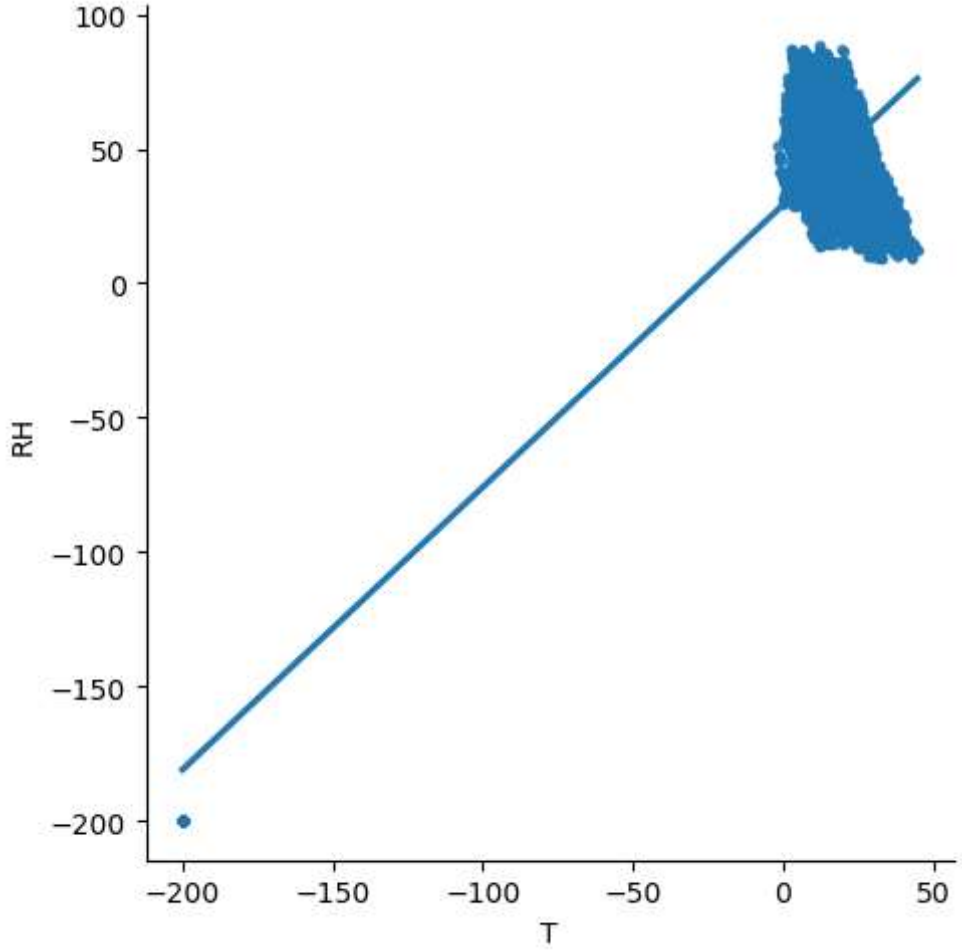

100
50
0
−50
−100
−150
−200
RH
−250
0
250
500
750
1000
1250
1500
NOx(GT)

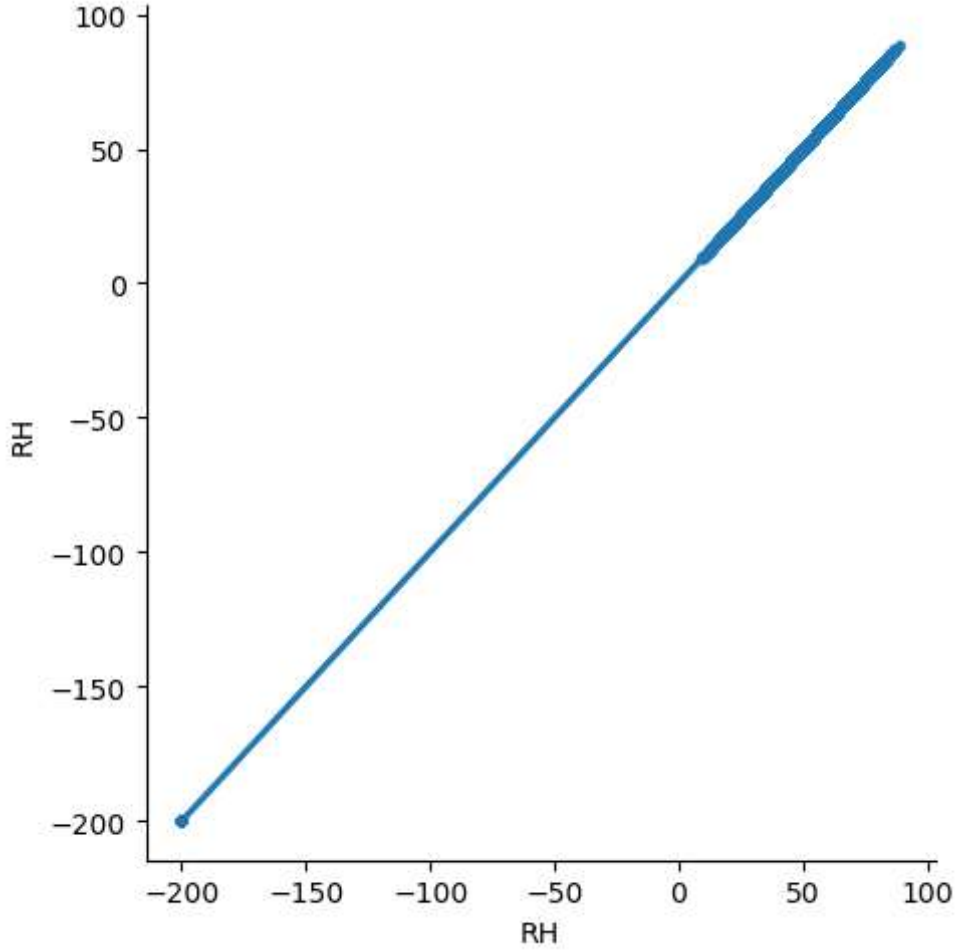

200
100
0
−100
−200
RH
0
500
1000
1500
2000
2500
PT08.S3(NOx)

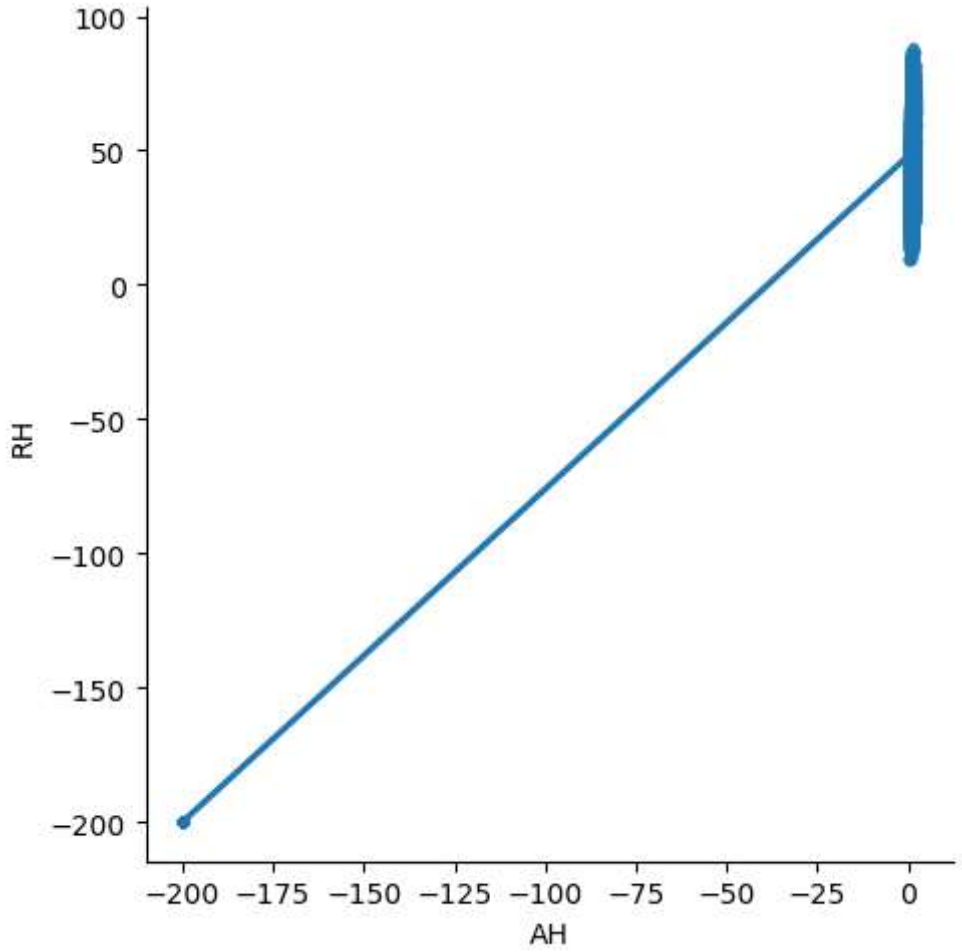

100
50
0
−50
−100
−150
−200
RH
−200
−100
0
100
200
300
NO2(GT)


150
100
50
0
−50
−100
−150
−200
RH
0
500
1000
1500
2000
2500
PT08.S4(NO2)


150
100
50
0
−50
−100
−150
−200
RH
0
500
1000
1500
2000
2500
PT08.S5(O3)


100
50
0
−50
−100
−150
−200
RH
−200
−150
−100
−50
0
50
T


100
50
0
−50
−100
−150
−200
RH
−200
−150
−100
−50
0
50
100
RH



In [9]:
```
from sklearn.preprocessing import StandardScaler          #import normalisation package
from sklearn.model_selection import train_test_split       #import train test split
from sklearn.linear_model import LinearRegression          #import linear regression package
from sklearn.metrics import mean_squared_error,mean_absolute_error   #import mean squared error and mean absolute error
```

In [10]:
```
X=data[col_].drop('RH',1)     #X-input features
y=data['RH']
```

In [11]:
```
ss=StandardScaler()     #initiatilise
```

```
X_std=ss.fit_transform(X)     #apply stardardisation
```

In [12]:
```
X_train, X_test, y_train, y_test=train_test_split(X_std,y,test_size=0.3, random_state=42)
```

In [13]:
```
print('Training data size:',X_train.shape)
print('Test data size:',X_test.shape)
```

```
Training data size: (6549, 12)
Test data size: (2808, 12)
```

In [14]:
```
lr=LinearRegression()
lr_model=lr.fit(X_train,y_train)          #fit the linear model on train data
print('Intercept:',lr_model.intercept_)
print('--------------------------------')
print('Slope:')
print(list(zip(X.columns.tolist(),lr_model.coef_)))
```

```
Intercept: 39.48586056471033
--------------------------------
Slope:
[('CO(GT)', 0.37631295673257303), ('PT08.S1(CO)', 3.2761957157732318), ('NMHC(GT)', -2.1453360996088615), ('C6H6(GT)', -48.1027730733
6059), ('PT08.S2(NMHC)', -16.703868112073955), ('NOx(GT)', 7.8025418758624285), ('PT08.S3(NOx)', -6.431116299980261), ('NO2(GT)', -6.
404494232464043), ('PT08.S4(NO2)', 27.603146537409394), ('PT08.S5(O3)', -0.16832955987058656), ('T', -87.61049424500278), ('AH', 175.
28393938691548)]
```

In [15]:
```
y_pred=lr_model.predict(X_test)                      #predict using the model
rmse=np.sqrt(mean_squared_error(y_test,y_pred))      #calculate rmse
print('RMSE of model:',rmse)
print("linear regression: ", lr_model.score(X_test, y_test))
```

```
RMSE of model: 7.879527482532906
linear regression:  0.9762585204595909
```

In [16]:
```
from sklearn.ensemble import RandomForestRegressor          #import random forest regressor
rf_reg=RandomForestRegressor()
```

In [17]:
```
rf_model=rf_reg.fit(X_train,y_train)         #fit model
y_pred_rf=rf_model.predict(X_test)           #predict
```

In [18]:
```
print('RMSE of predicted RH in RF model:',np.sqrt(mean_squared_error(y_test,y_pred_rf)))
print("Random Forest: ", rf_model.score(X_test, y_test))
```

```
RMSE of predicted RH in RF model: 0.6025343247779432
Random Forest:  0.9998611739342902
```

In [ ]:

In [ ]: