pyspark air polution projs

June 30, 2023

1 Air Quality Analysis using PySpark

This project focuses on analyzing air quality data using PySpark and Pandas, two popular libraries in the field of data processing and analysis. The dataset used for this project is the "Air Quality Dataset Hourly averaged responses from an array of 5 metal oxide chemical sensors" available on Kaggle.

1.1 Dataset Description

The "Air Quality Dataset Hourly averaged responses from an array of 5 metal oxide chemical sensors" contains hourly averaged responses from an array of 5 metal oxide chemical sensors embedded in an Air Quality Chemical Multisensor Device. The data was collected in a controlled chamber with a gas analyser reference device and 5 metal oxide chemical sensors. The goal is to estimate the relative humidity and temperature values based on the response of the chemical sensors.

0) Date (DD/MM/YYYY)
1) Time (HH.MM.SS)
2) True hourly averaged concentration CO in mg/m^3 (reference analyzer)
3) PT08.S1 (tin oxide) hourly averaged sensor response (nominally CO targeted)
4) True hourly averaged overall Non Metanic HydroCarbons concentration in microg/m^3 (reference analyzer)
5) True hourly averaged Benzene concentration in microg/m^3 (reference analyzer)
6) PT08.S2 (titania) hourly averaged sensor response (nominally NMHC targeted)
7) True hourly averaged NOx concentration in ppb (reference analyzer)
8) PT08.S3 (tungsten oxide) hourly averaged sensor response (nominally NOx targeted)
9) True hourly averaged NO2 concentration in microg/m^3 (reference analyzer)
10) PT08.S4 (tungsten oxide) hourly averaged sensor response (nominally NO2 targeted)
11) PT08.S5 (indium oxide) hourly averaged sensor response (nominally O3 targeted)
12) Temperature in Â°C
13) Relative Humidity (%)
14) AH Absolute Humidity

More Notes:

- GT = Global Throposphere

1.2 Project Overview

This project follows a modular approach, with separate files for data preprocessing, data analysis, and model development. The main focus is on utilizing PySpark to preprocess the data, perform

exploratory data analysis, and build a machine learning model for predicting air quality parameters.

The project involves the following steps:

1. **Data Preprocessing**: In this step, the raw dataset is cleaned, transformed, and prepared for further analysis. Data cleaning involves handling missing values, removing duplicates, and addressing any inconsistencies in the data. Feature engineering techniques may be applied to extract relevant information.
2. **Data Analysis**: Once the data is preprocessed, exploratory data analysis techniques are applied using Pandas and Plotly. Various statistical measures, visualizations, and insights are derived to understand the characteristics and patterns present in the dataset. This step helps in gaining a deeper understanding of the air quality data.
3. **Model Development**: After the data analysis, a machine learning model is developed using PySpark. The model is trained on the preprocessed dataset to predict air quality parameters such as relative humidity and temperature. Different algorithms, such as regression or classification models, can be explored and evaluated based on their performance metrics.

By following this modular approach, the project aims to provide a comprehensive understanding of the air quality dataset and enable accurate predictions of air quality parameters using machine learning techniques.

Note: The code for this project, including the data preprocessing, analysis, and model development, can be found in the respective files in the project repository.

[ ]:
```
import pandas as pd
import numpy as np
import datetime as dt

from pyspark.sql import SparkSession
```

[ ]:
```
spark = SparkSession.builder\
    .master('local[*]')\
    .appName('air_polution')\
    .getOrCreate()
```

[ ]:
```
df = spark.read.csv('datasets/Air Quality.csv', inferSchema=True, header=True)
df.show(5)
```

```
+----------+-------------------+------+-----------+--------+--------+-----------
--+-------+------------+-------+------------+-----------+----+----+------+----+-
---+
|      Date|               Time|CO(GT)|PT08_S1(CO)|NMHC(GT)|C6H6(GT)|PT08_S2(NMH
C)|NOx(GT)|PT08_S3(NOx)|NO2(GT)|PT08_S4(NO2)|PT08_S5(O3)|   T|  RH|
AH|_c15|_c16|
+----------+-------------------+------+-----------+--------+--------+-----------
--+-------+------------+-------+------------+-----------+----+----+------+----+-
---+
|10/03/2004|2023-06-30 18:00:00|   2.6|       1360|     150|    11.9|
1046|    166|        1056|    113|        1692|
```

```
1268|13.6|48.9|0.7578|null|null|
|10/03/2004|2023-06-30 19:00:00|   2.0|       1292|     112|     9.4|
955|    103|        1174|     92|        1559|
972|13.3|47.7|0.7255|null|null|
|10/03/2004|2023-06-30 20:00:00|   2.2|       1402|      88|     9.0|
939|    131|        1140|    114|        1555|
1074|11.9|54.0|0.7502|null|null|
|10/03/2004|2023-06-30 21:00:00|   2.2|       1376|      80|     9.2|
948|    172|        1092|    122|        1584|
1203|11.0|60.0|0.7867|null|null|
|10/03/2004|2023-06-30 22:00:00|   1.6|       1272|      51|     6.5|
836|    131|        1205|    116|        1490|
1110|11.2|59.6|0.7888|null|null|
+----------+-------------------+------+-----------+--------+--------+-----------
--+-------+------------+-------+------------+-----------+----+----+------+----+-
---+
only showing top 5 rows
```

[ ]:
```
df = df.drop(*['_c15','_c16'])
```

[ ]:
```
df.printSchema()
```

```
root
 |-- Date: string (nullable = true)
 |-- Time: timestamp (nullable = true)
 |-- CO(GT): double (nullable = true)
 |-- PT08_S1(CO): integer (nullable = true)
 |-- NMHC(GT): integer (nullable = true)
 |-- C6H6(GT): double (nullable = true)
 |-- PT08_S2(NMHC): integer (nullable = true)
 |-- NOx(GT): integer (nullable = true)
 |-- PT08_S3(NOx): integer (nullable = true)
 |-- NO2(GT): integer (nullable = true)
 |-- PT08_S4(NO2): integer (nullable = true)
 |-- PT08_S5(O3): integer (nullable = true)
 |-- T: double (nullable = true)
 |-- RH: double (nullable = true)
 |-- AH: double (nullable = true)
```

2 Data Transform

[ ]:
```
CONSTANT_MODULE_FILE = "module/airpolution_data_constant.py"
```

[ ]:
```
%%writefile {CONSTANT_MODULE_FILE}
```

```
LABEL_KEY = "T"
FEATURE_KEY = [
 'CO(GT)',
 'PT08_S1(CO)',
 'NMHC(GT)',
 'C6H6(GT)',
 'PT08_S2(NMHC)',
 'NOx(GT)',
 'PT08_S3(NOx)',
 'NO2(GT)',
 'PT08_S4(NO2)',
 'PT08_S5(O3)',
 'RH',
 'AH']

ALL_KEY = [
 'CO(GT)',
 'PT08_S1(CO)',
 'NMHC(GT)',
 'C6H6(GT)',
 'PT08_S2(NMHC)',
 'NOx(GT)',
 'PT08_S3(NOx)',
 'NO2(GT)',
 'PT08_S4(NO2)',
 'PT08_S5(O3)',
 'RH',
 'AH',
 'T']

def transformed_name(key):
    """Rename transformed features"""

    return key + "_tn"

def vectorize_name(key):
    """"Rename vectorize features"""

    return key + "_Vect"
```

```
Overwriting module/airpolution_data_constant.py
```

[ ]:
```
TRANSFORM_MODULE_FILE = "module/airpolution_data_transform.py"
```

[ ]:
```
%%writefile {TRANSFORM_MODULE_FILE}

import airpolution_data_constant
```

```
from pyspark.sql.functions import concat, date_format, col, to_timestamp, lit,␣
 ↪to_date, when, mean, coalesce, avg, log
from pyspark.ml.feature import MinMaxScaler, VectorAssembler
from pyspark.ml import Pipeline
from pyspark.sql.types import DoubleType
from pyspark.sql.functions import udf
import numpy as np

_FEATURE_KEY = airpolution_data_constant.FEATURE_KEY
_LABEL_KEY = airpolution_data_constant.LABEL_KEY
_ALL_KEY = airpolution_data_constant.ALL_KEY

_transformed_name = airpolution_data_constant.transformed_name
_vectorize_name = airpolution_data_constant.vectorize_name


def change_time_format(inputs):
  """This Function Changing the time format"""

  df = inputs.withColumn('Time', date_format('Time', 'HH:mm:ss'))

  return df

def merge_date_time(inputs):
  """This Function Merging Date & Time Column"""

  datetime_col = concat(inputs.Date, lit(" "), inputs.Time)
  df = inputs.withColumn("datetime", to_timestamp(datetime_col, "dd/MM/yyyy HH:
 ↪mm:ss"))

  return df

def clean_date_format(inputs):
  """This Function Fix Date & Time Type"""

  df = inputs.withColumn("Date", to_date("Date", "dd/MM/yyyy"))

  return df

def clean_outlier(inputs):
  """This Function clean outlier and fill it with mean"""

  df = inputs.replace(-200, None)

  for i in df.columns:
    if i not in ["Date","Time","datetime"]:
```

```
        mean_col_value = df.select(mean(col(i))).collect()[0][0]

        df = df.na.fill(mean_col_value, i)

    return df

def drop_duplicate(inputs):
    """This Function deleting duplicate value in dataframe"""

    df = inputs.dropDuplicates()

    return df

def data_distribution(inputs):
    """This function transforms the data distribution"""

    column_transform_type = {
        'CO(GT)': 'Log',
        'PT08_S1(CO)': 'Reciprocal',
        'NMHC(GT)': 'Log',
        'C6H6(GT)': 'Log',
        'PT08_S2(NMHC)': 'Log',
        'NOx(GT)': 'Log',
        'PT08_S3(NOx)': 'Log',
        'NO2(GT)': 'Original',
        'PT08_S4(NO2)': 'Original',
        'PT08_S5(O3)': 'Log',
        'T': 'Original',
        'RH': 'Original',
        'AH': 'Original'
    }

    transformed_df = inputs

    for col_name, transform_type in column_transform_type.items():
        if transform_type == "Log":
            transformed_df = transformed_df.withColumn(col_name,␣
 ↪log(col(col_name)))
        elif transform_type == "Reciprocal":
            transformed_df = transformed_df.withColumn(col_name, 1 /␣
 ↪col(col_name))

    return transformed_df

def normalize_data(inputs):
    """Normalize the data"""
```

```
    column_to_remove = ['Date', 'Time', 'datetime']
    column_to_use = [i for i in inputs.columns if i not in column_to_remove]

    df = inputs

    for column in column_to_use:
        assembler = VectorAssembler(inputCols=[column],␣
 ↪outputCol=_vectorize_name(column))
        minmax = MinMaxScaler(inputCol=_vectorize_name(column),␣
 ↪outputCol=_transformed_name(column))
        pipeline = Pipeline(stages=[assembler, minmax])
        df = pipeline.fit(df).transform(df).drop(_vectorize_name(column))

    return df

def preprocessing_fn(inputs, dist_transform=False, normalize=False):
  """Main Function"""

  df = change_time_format(inputs)
  df = merge_date_time(df)
  df = clean_date_format(df)
  df = clean_outlier(df)
  df = drop_duplicate(df)

  if dist_transform:
    df = data_distribution(df)

  if normalize:
    df = normalize_data(df)

  return df
```

```
Overwriting module/airpolution_data_transform.py
```

Transform file do cleaning the null data, outlier data, time format, normalizing data and distribution transform that will be feed for the model development.

[ ]:
```
import sys

sys.path.append("C:\\Users\\Mario\\Documents\\data analyst␣
 ↪projs\\pyspark_airpolution\\module")
```

[ ]:
```
import airpolution_data_transform

df_processing = airpolution_data_transform.preprocessing_fn(df)
```

[ ]:
```
df_processing.describe().show()
```

```
+-------+--------+------------------+------------------+-----------------+------
```

```
-----------+-----------------+-----------------+-----------------+-----------
-------+-----------------+-----------------+-----------------+---------------
--+-------------------+
|summary|    Time|            CO(GT)|       PT08_S1(CO)|         NMHC(GT)|
C6H6(GT)|    PT08_S2(NMHC)|           NOx(GT)|      PT08_S3(NOx)|
NO2(GT)|      PT08_S4(NO2)|       PT08_S5(O3)|                T|
RH|                 AH|
+-------+--------+-----------------+-----------------+-----------------+------
------------+-----------------+-----------------+-----------------+-----------
-------+-----------------+-----------------+-----------------+---------------
--+-------------------+
|  count|    9357|             9357|             9357|             9357|
9357|             9357|             9357|             9357|              9357|
9357|              9357|             9357|              9357|
9357|
|   mean|    null| 2.152749543914555|1099.8005771080475|218.0792989205942|10.083
105327549735|939.1473762958213|246.73966014748316| 835.4742973175163|113.0752377
8988993|1456.2542481564603|1022.8706850486267|18.31782894005132|
49.23420086753411| 1.0255302747191637|
| stddev|    null|1.3160683129140567| 212.7917331700746|63.87068350364549|
7.302650251426652|261.5602377875098|193.42693239497223|251.74397207545198|
43.92096782111363| 339.3675631184069|
390.612363262573|8.657639349901688|16.974801298982392|0.39583538239942756|
|    min|00:00:00|              0.1|              647|                7|
0.1|             383|                2|              322|                 2|
551|              221|             -1.9|              9.2|             0.1847|
|    max|23:00:00|             11.9|             2040|             1189|
63.7|             2214|             1479|             2683|              340|
2775|              2523|             44.6|             88.7|
2.231|
+-------+--------+-----------------+-----------------+-----------------+------
------------+-----------------+-----------------+-----------------+-----------
-------+-----------------+-----------------+-----------------+---------------
--+-------------------+
```

[ ]:
```
df_processing.printSchema()
```

```
root
 |-- Date: date (nullable = true)
 |-- Time: string (nullable = true)
 |-- CO(GT): double (nullable = false)
 |-- PT08_S1(CO): integer (nullable = true)
 |-- NMHC(GT): integer (nullable = true)
 |-- C6H6(GT): double (nullable = false)
 |-- PT08_S2(NMHC): integer (nullable = true)
 |-- NOx(GT): integer (nullable = true)
 |-- PT08_S3(NOx): integer (nullable = true)
```

```
 |-- NO2(GT): integer (nullable = true)
 |-- PT08_S4(NO2): integer (nullable = true)
 |-- PT08_S5(O3): integer (nullable = true)
 |-- T: double (nullable = false)
 |-- RH: double (nullable = false)
 |-- AH: double (nullable = false)
 |-- datetime: timestamp (nullable = true)
```

[ ]:
```
df_processing.show()
```

```
+----------+--------+-----------------+-----------+--------+--------+----------
---+-------+------------+-------+------------+-----------+----+----+------+-----
--------------+
|      Date|    Time|            CO(GT)|PT08_S1(CO)|NMHC(GT)|C6H6(GT)|PT08_S2(NM
HC)|NOx(GT)|PT08_S3(NOx)|NO2(GT)|PT08_S4(NO2)|PT08_S5(O3)|   T|  RH|    AH|
datetime|
+----------+--------+-----------------+-----------+--------+--------+----------
---+-------+------------+-------+------------+-----------+----+----+------+-----
--------------+
|2004-03-11|07:00:00|              1.1|       1144|      29|     3.2|
667|     98|        1490|     82|        1339|
730|10.2|59.6|0.7417|2004-03-11 07:00:00|
|2004-03-14|14:00:00|              1.8|       1207|      84|     7.5|
879|    103|        1104|    102|        1490|
872|21.4|30.2|0.7616|2004-03-14 14:00:00|
|2004-03-15|09:00:00|              8.1|       1961|     618|    36.7|
1701|    478|         537|    149|        2665|
2184|14.8|54.3|0.9076|2004-03-15 09:00:00|
|2004-03-19|22:00:00|              2.2|       1175|     218|     9.1|
945|    143|         904|    116|        1604|
1081|14.7|57.6|0.9573|2004-03-19 22:00:00|
|2004-03-22|17:00:00|              2.6|       1152|     185|    12.4|
1062|    138|         928|    103|        1606|
850|20.2|28.5|0.6682|2004-03-22 17:00:00|
|2004-03-24|22:00:00|              1.7|       1047|      97|     5.9|
809|    118|        1064|    108|        1435|
766|10.1|66.9|0.8248|2004-03-24 22:00:00|
|2004-04-08|04:00:00|2.1527495439145157|        823|      38|     1.8|
568|     43|        1366|     57|        1263|        699|
8.3|75.6|0.8302|2004-04-08 04:00:00|
|2004-04-29|01:00:00|              1.0|       1042|      84|     5.2|
772|     51|         923|     61|        1495|
968|16.8|59.5|1.1269|2004-04-29 01:00:00|
|2004-04-29|08:00:00|              7.2|       1771|    1129|    36.2|
1690|    368|         461|    125|        2572|
1982|16.1|56.5|1.0274|2004-04-29 08:00:00|
|2004-05-03|03:00:00|              0.4|        872|     218|     1.7|
```

```
564|    246|        1284|    113|        1385|
494|15.6|65.3|1.1484|2004-05-03 03:00:00|
|2004-05-06|10:00:00|               2.5|       1043|     218|    12.5|
1066|    161|         789|    114|        1692|
1014|20.2|37.7|0.8833|2004-05-06 10:00:00|
|2004-05-09|17:00:00|2.1527495439145157|        968|     218|     5.4|
785|    246|        1011|    113|        1446|
550|17.8|44.4|0.8967|2004-05-09 17:00:00|
|2004-05-15|02:00:00|               1.0|        935|     218|     5.4|
786|     55|         964|     71|        1389|
954|17.4|43.4|0.8553|2004-05-15 02:00:00|
|2004-06-05|12:00:00|               1.4|        914|     218|     8.0|
900|     86|        1010|     68|        1596|
706|30.7|26.0|1.1287|2004-06-05 12:00:00|
|2004-06-07|04:00:00|2.1527495439145157|        726|     218|     2.1|
591|     22|        1263|     34|        1386|        579|17.2|57.9|
1.127|2004-06-07 04:00:00|
|2004-06-21|13:00:00|               1.2|        838|     218|     7.2|
868|     66|        1005|     76|        1454|
540|30.1|18.5|0.7747|2004-06-21 13:00:00|
|2004-06-29|17:00:00|               3.7|       1317|     218|    21.5|
1338|    200|         593|    198|        2132|
1362|37.1|26.9|1.6635|2004-06-29 17:00:00|
|2004-07-04|13:00:00|               0.7|        884|     218|     3.9|
703|     34|        1076|     46|        1394|        488|37.9|18.8|
1.222|2004-07-04 13:00:00|
|2004-07-08|02:00:00|               0.6|        962|     218|     5.8|
803|     43|         853|     60|        1507|       1066|27.5|34.0|
1.231|2004-07-08 02:00:00|
|2004-07-12|05:00:00|               0.5|        835|     218|     3.3|
672|     43|        1053|     43|        1423|
628|20.0|51.8|1.1967|2004-07-12 05:00:00|
+----------+--------+------------------+-----------+--------+--------+----------
---+-------+-----------+-------+-----------+-----------+----+----+------+-----
--------------+
only showing top 20 rows
```

3 Data Analysist

3.1 Importing Library

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

from itertools import cycle
import plotly.graph_objects as go
```

```
import plotly.express as px
from plotly.subplots import make_subplots

%matplotlib inline
```

[ ]:
```
analys_df = df_processing.toPandas()
```

```
c:\Users\Mario\anaconda3\envs\bigdata_env\lib\site-
packages\pyspark\sql\pandas\conversion.py:251: FutureWarning: Passing unit-less
datetime64 dtype to .astype is deprecated and will raise in a future version.
Pass 'datetime64[ns]' instead
  series = series.astype(t, copy=False)
```

[ ]:
```
analys_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9357 entries, 0 to 9356
Data columns (total 16 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   int32
 4   NMHC(GT)        9357 non-null   int32
 5   C6H6(GT)        9357 non-null   float64
 6   PT08_S2(NMHC)   9357 non-null   int32
 7   NOx(GT)         9357 non-null   int32
 8   PT08_S3(NOx)    9357 non-null   int32
 9   NO2(GT)         9357 non-null   int32
 10  PT08_S4(NO2)    9357 non-null   int32
 11  PT08_S5(O3)     9357 non-null   int32
 12  T               9357 non-null   float64
 13  RH              9357 non-null   float64
 14  AH              9357 non-null   float64
 15  datetime        9357 non-null   datetime64[ns]
dtypes: datetime64[ns](1), float64(5), int32(8), object(2)
memory usage: 877.3+ KB
```

[ ]:
```
analys_df.sort_values(['datetime'])
```

[ ]:

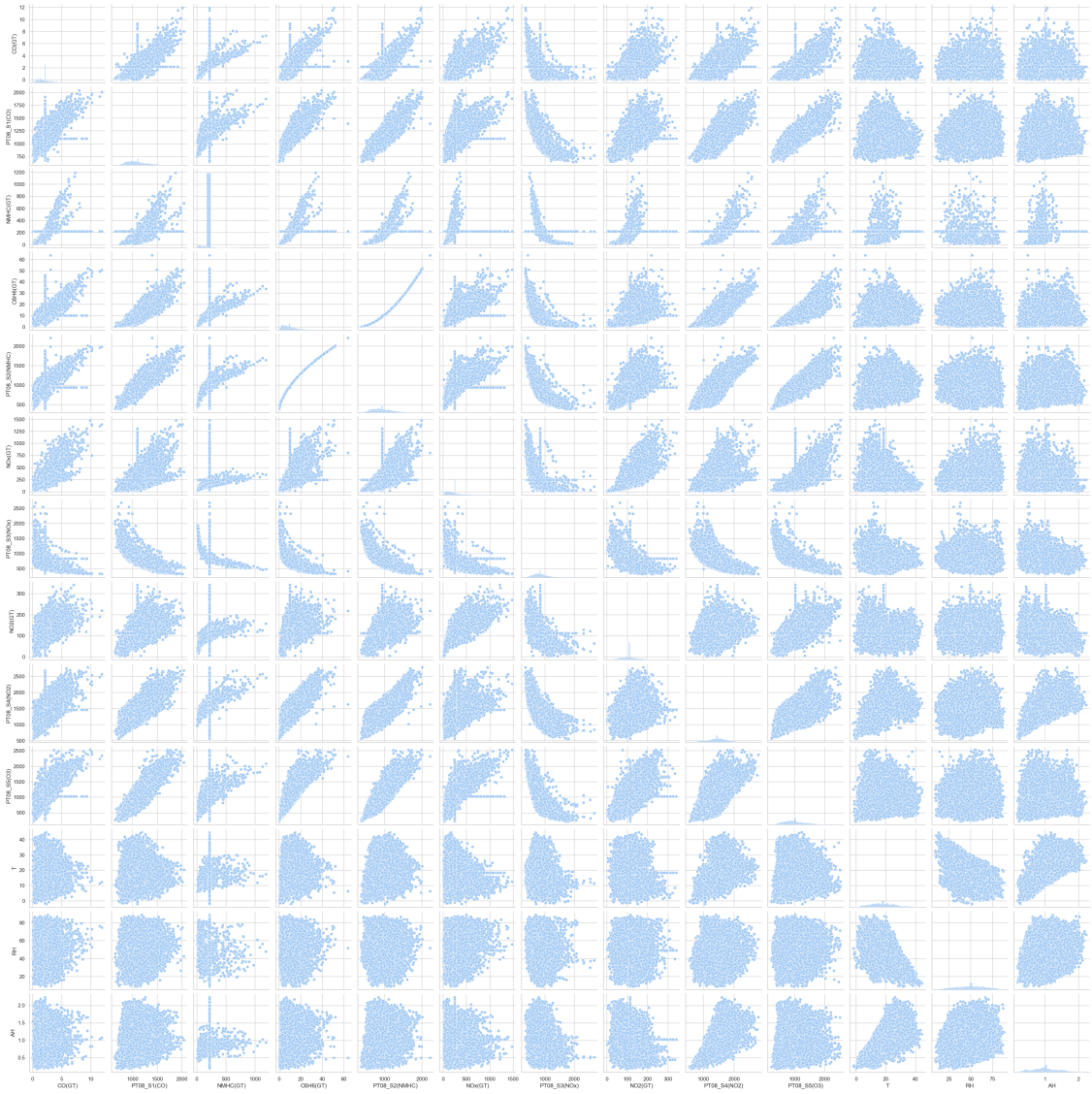
	Date	Time	CO(GT)	PT08_S1(CO)	NMHC(GT)	C6H6(GT)	\
2504	2004-03-10	18:00:00	2.6	1360	150	11.9	
2458	2004-03-10	19:00:00	2.0	1292	112	9.4	
5009	2004-03-10	20:00:00	2.2	1402	88	9.0	
6720	2004-03-10	21:00:00	2.2	1376	80	9.2	
5069	2004-03-10	22:00:00	1.6	1272	51	6.5	
…	…	…	…	…	…	…	

```
499   2005-04-04  10:00:00     3.1          1314       218       13.5
9007  2005-04-04  11:00:00     2.4          1163       218       11.4
1237  2005-04-04  12:00:00     2.4          1142       218       12.4
6265  2005-04-04  13:00:00     2.1          1003       218        9.5
828   2005-04-04  14:00:00     2.2          1071       218       11.9

      PT08_S2(NMHC)  NOx(GT)  PT08_S3(NOx)  NO2(GT)  PT08_S4(NO2)  \
2504           1046      166          1056      113          1692
2458            955      103          1174       92          1559
5009            939      131          1140      114          1555
6720            948      172          1092      122          1584
5069            836      131          1205      116          1490
…               …        …            …         …            …
499            1101      472           539      190          1374
9007           1027      353           604      179          1264
1237           1063      293           603      175          1241
6265            961      235           702      156          1041
828            1047      265           654      168          1129

      PT08_S5(O3)     T    RH      AH             datetime
2504         1268  13.6  48.9  0.7578  2004-03-10 18:00:00
2458          972  13.3  47.7  0.7255  2004-03-10 19:00:00
5009         1074  11.9  54.0  0.7502  2004-03-10 20:00:00
6720         1203  11.0  60.0  0.7867  2004-03-10 21:00:00
5069         1110  11.2  59.6  0.7888  2004-03-10 22:00:00
…             …    …     …      …                    …
499          1729  21.9  29.3  0.7568  2005-04-04 10:00:00
9007         1269  24.3  23.7  0.7119  2005-04-04 11:00:00
1237         1092  26.9  18.3  0.6406  2005-04-04 12:00:00
6265          770  28.3  13.5  0.5139  2005-04-04 13:00:00
828           816  28.5  13.1  0.5028  2005-04-04 14:00:00

[9357 rows x 16 columns]
```

3.2 Pairplot Analysis

[ ]:
```
sns.set_theme(style="whitegrid", palette="pastel")
sns.pairplot(analys_df)
```

[ ]:
```
<seaborn.axisgrid.PairGrid at 0x26d128a1100>
```

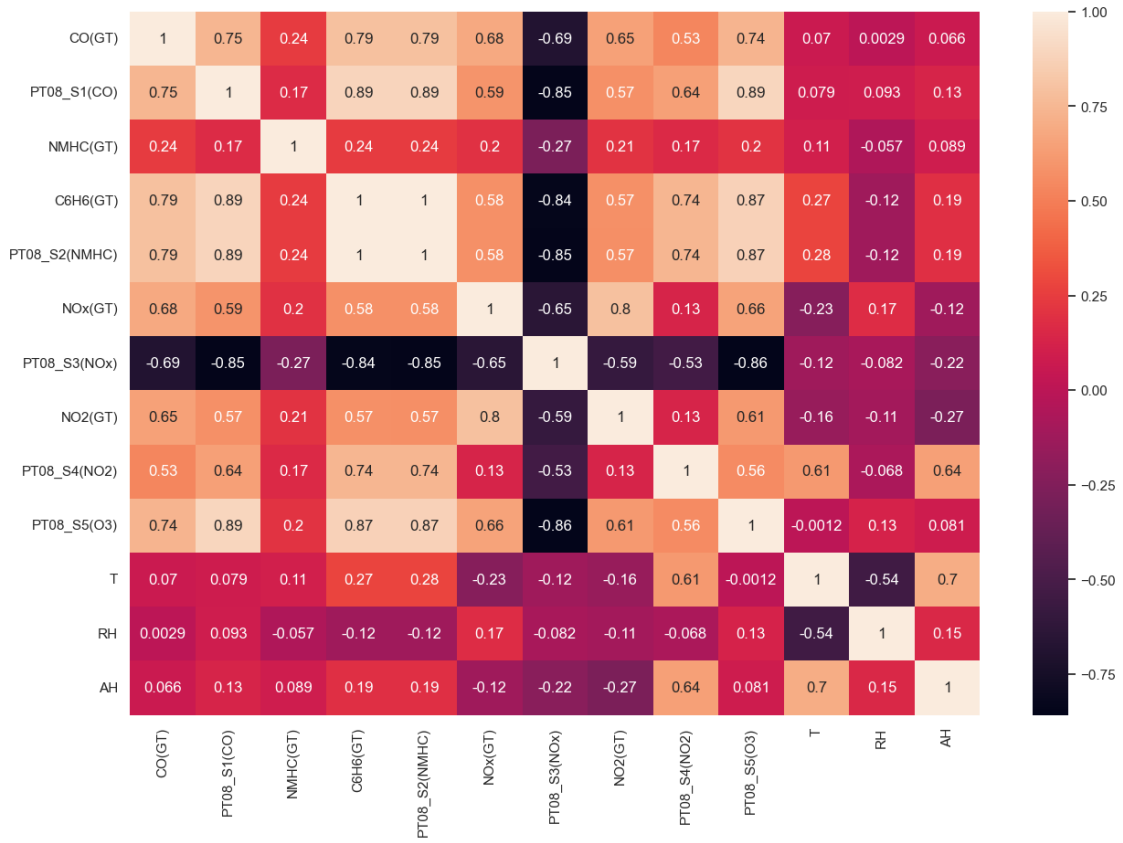


From the pairplot we can see there are some data that looks flat like NMHC(GT), that flat data are the outlier data that already been change with the mean value, since the outlier has being plot againts each other column the result is give us the flat data that we can look in the pairplot.

[ ]:
```
plt.figure(figsize=(15,10))
sns.heatmap(analys_df.corr(method="spearman"), annot=True)
```

```
C:\Users\Mario\AppData\Local\Temp\ipykernel_5268\2897492761.py:2: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future
version, it will default to False. Select only valid columns or specify the
value of numeric_only to silence this warning.
  sns.heatmap(analys_df.corr(method="spearman"), annot=True)
```

[ ]: <Axes: >



From the pairplot above we know that:

- Benzene has correlation with another columns except CO(GT), NOx, T, RH, AH
- Carbon monokside has positive correlation with Benzene in the scale of global troposphere
- The only chemical measurements that has positive correlation is CO(GT) and PT08_S1 as the global troposphere distribute the data linear the same as the sensors data distributed
- Data in columns T, RH, AH does not show significant correlation to another columns
- using the pairplot we can know that the data has more left skewed distributed data.

3.3 Forecast Analysis

[ ]:
```
df_monthly_avg = analys_df.groupby(analys_df['datetime'].dt.to_period('M')).
  ↪mean()
df_monthly_avg = df_monthly_avg.reset_index()
df_monthly_avg['datetime'] = df_monthly_avg['datetime'].dt.strftime('%Y-%m')
```

```
C:\Users\Mario\AppData\Local\Temp\ipykernel_5268\2822327006.py:1: FutureWarning:
The default value of numeric_only in DataFrameGroupBy.mean is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only
or select only columns which should be valid for the function.
  df_monthly_avg =
analys_df.groupby(analys_df['datetime'].dt.to_period('M')).mean()
```

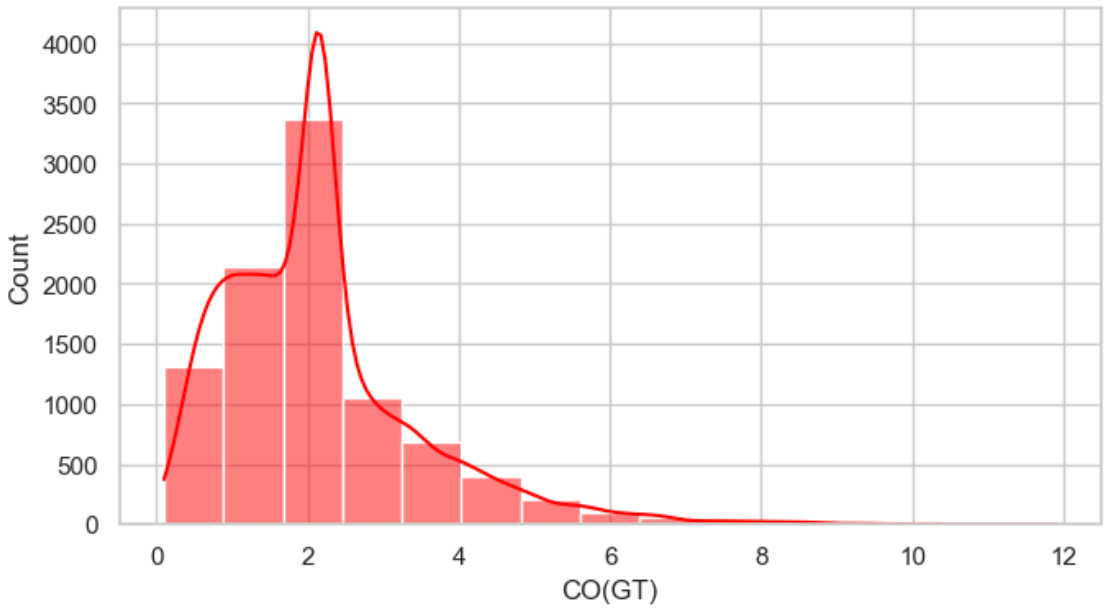
[ ]:
```
fig = px.line(df_monthly_avg, x="datetime", y=df_monthly_avg.columns,␣
↪title='Avg Monthly vs Variabels')
fig.show()
```

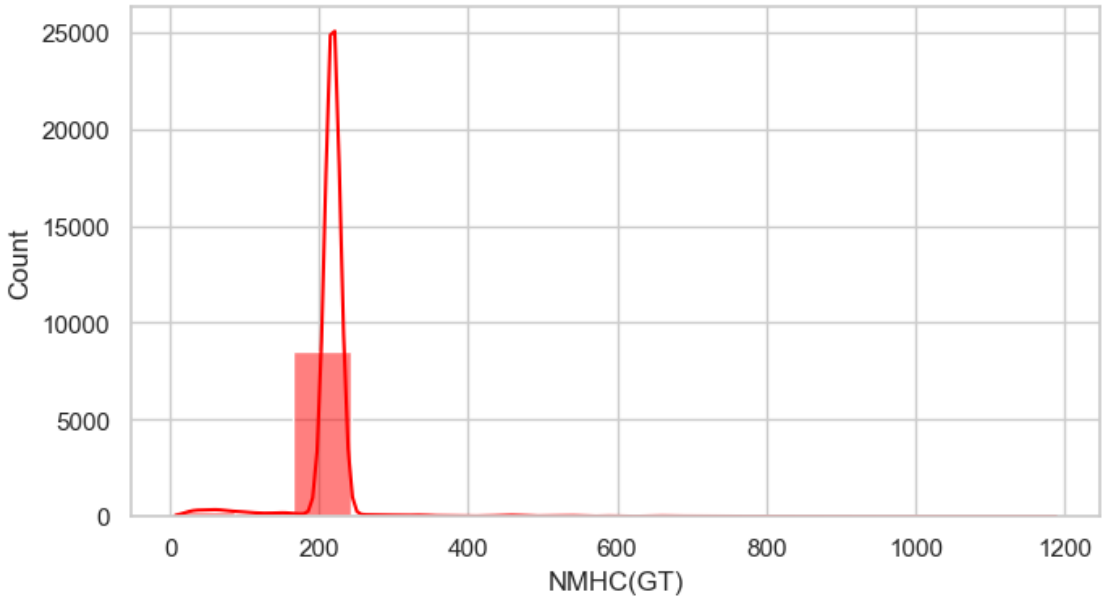
3.3.1 monthly forecast average variables

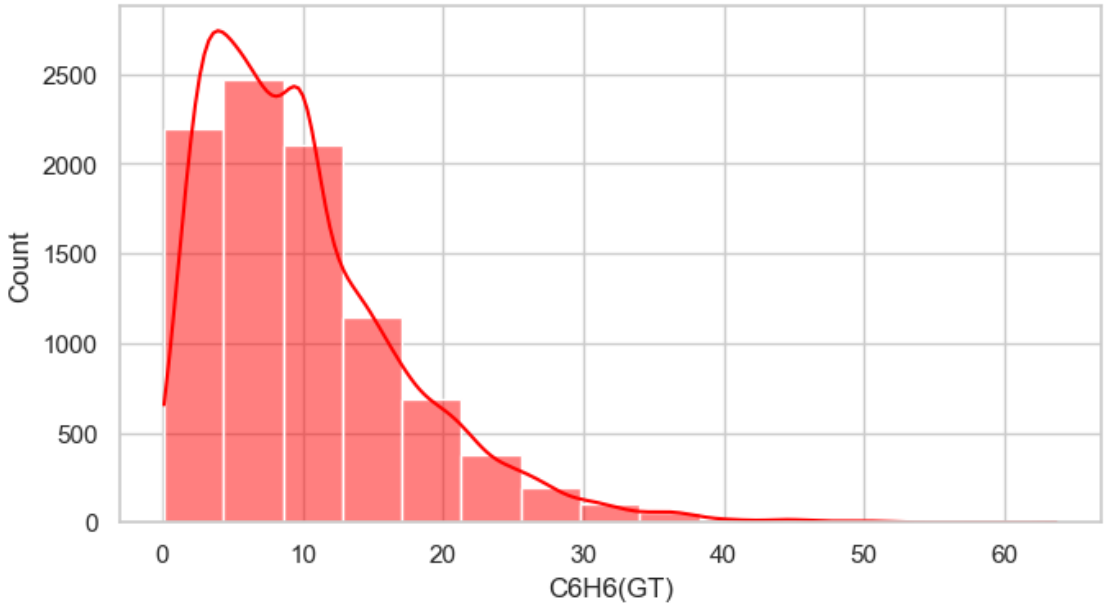
- from the forecast above we know that NOx (Nitric Oxide) has higher value compared to other variables.
- The highest value of benzene happend in oct 2004.
- The highest value of carbon monoksida happend in dec 2004.
- The highest value of Non Metanic HydroCarbons happend in April 2004.
- the highest value of Nitrogen Oksida happend in Nov 2004.
- The highest value of Nitrogen Dioksida happend in Feb 2005.

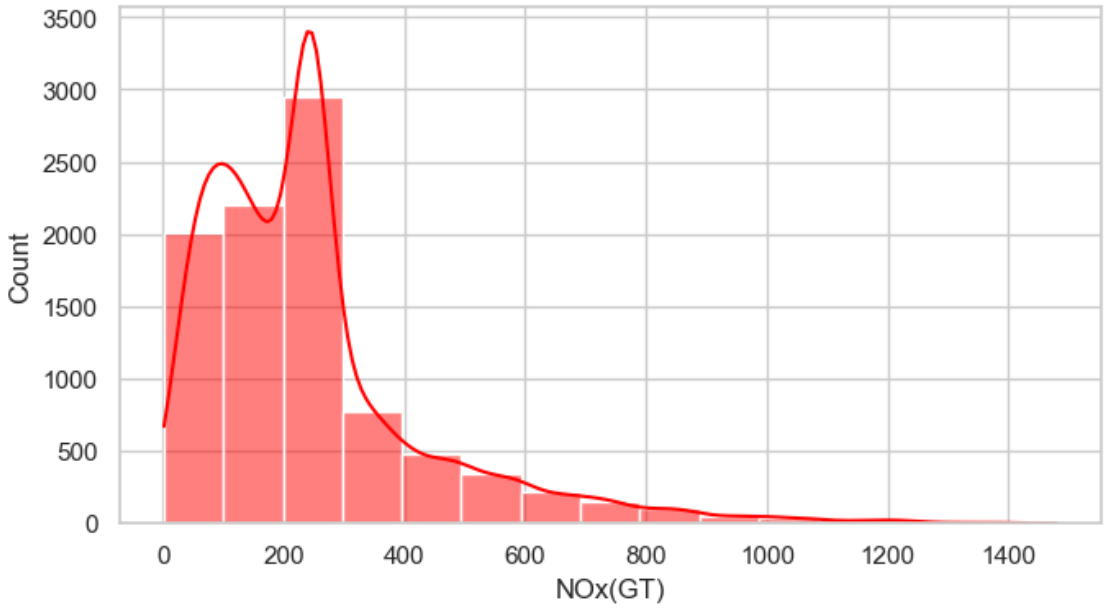
3.4 Checking data distributed

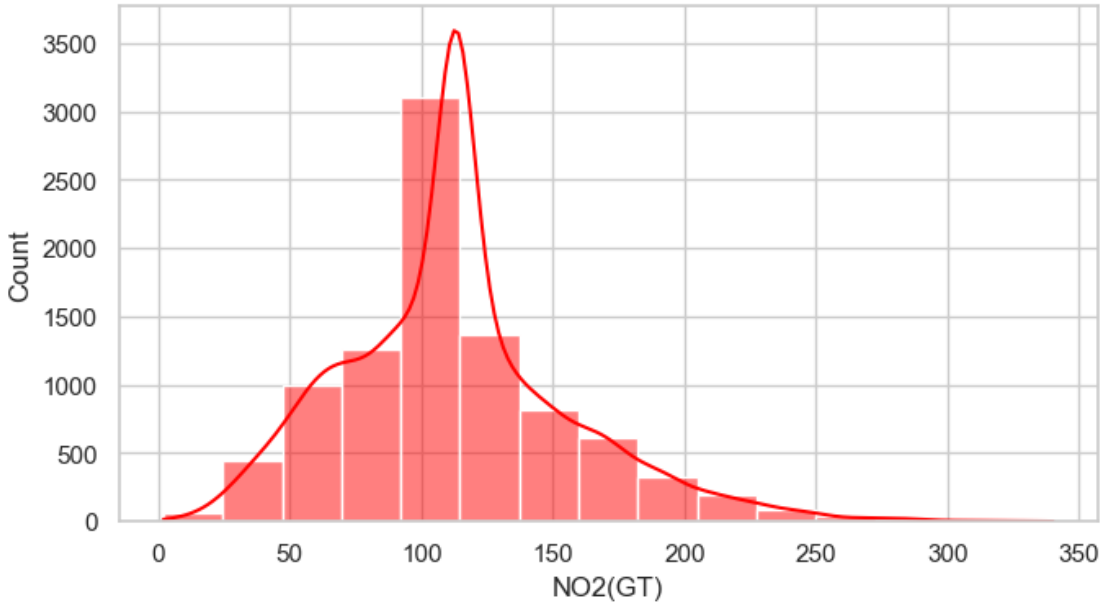
[ ]:
```
for i in analys_df.columns.drop(['Date','Time','datetime']):
    if "GT" in i:
        plt.figure(figsize=(17,9))
        plt.subplot(2,2,1)
        sns.
↪histplot(x=analys_df[i],stat="count",color="red",bins=15,kde={'alpha':0.5})
```




25000
20000
15000
10000
5000
0
Count
0
200
400
600
800
1000
1200
NMHC(GT)


2500
2000
1500
1000
500
0
Count
0
10
20
30
40
50
60
C6H6(GT)



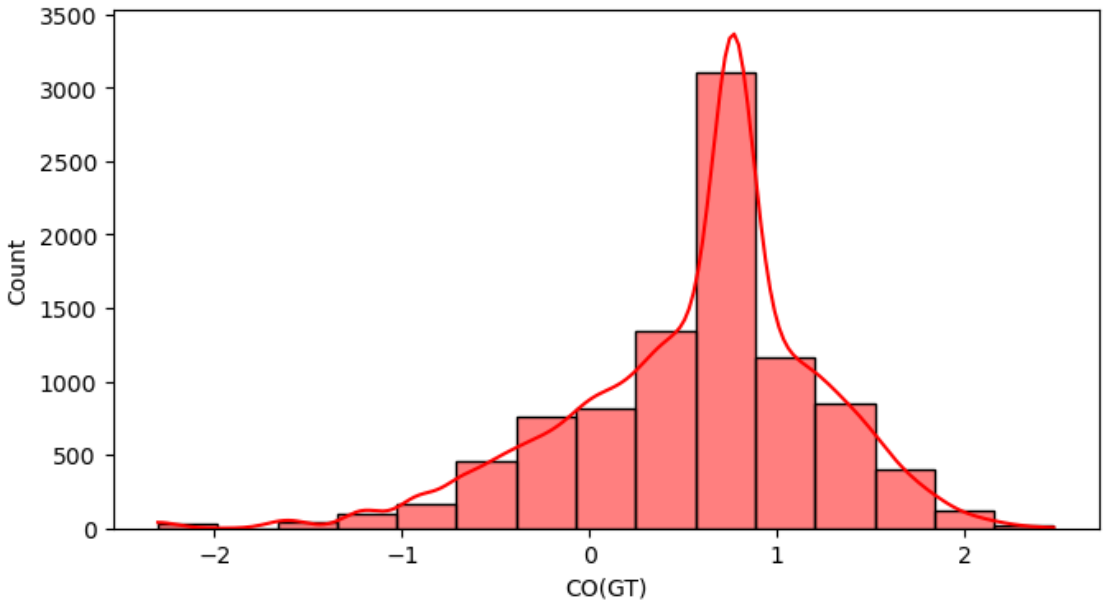


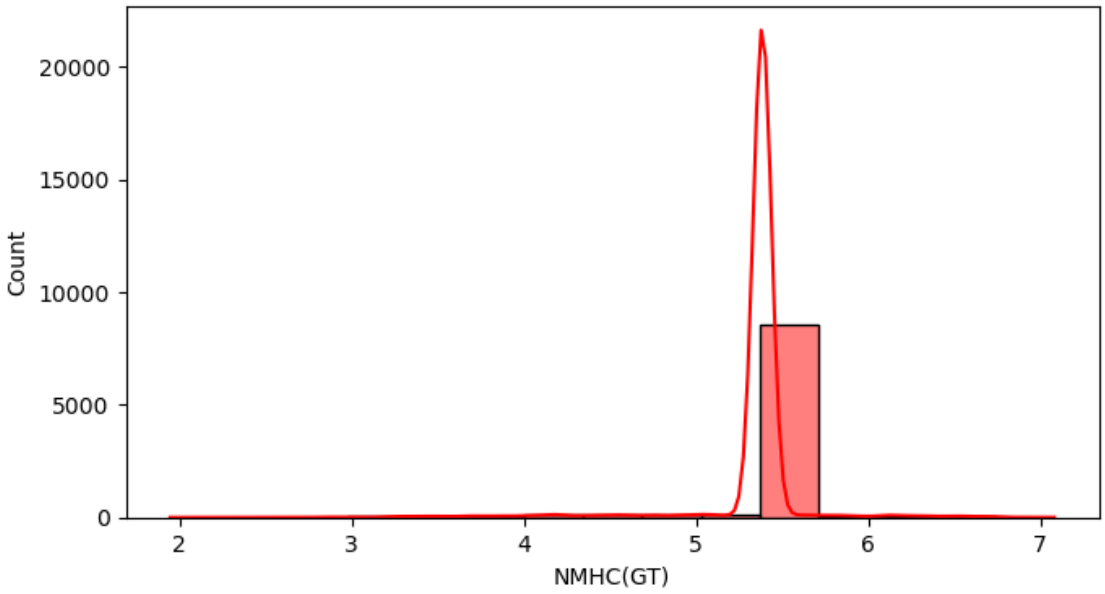
From the histplot above we know that Nitro Oksida, Benzene, Carbon Monoksida has left skewed data.

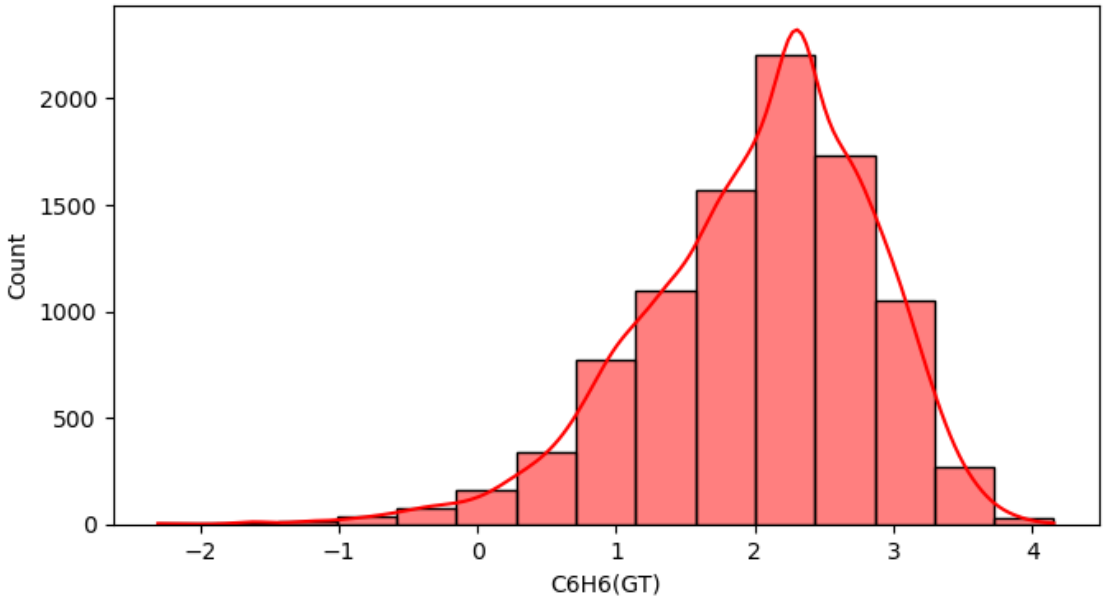
[ ]:
```
analys_df_log = analys_df[analys_df.columns.drop(['Date','Time','datetime'])].
↪apply(lambda x: np.log(x))
```

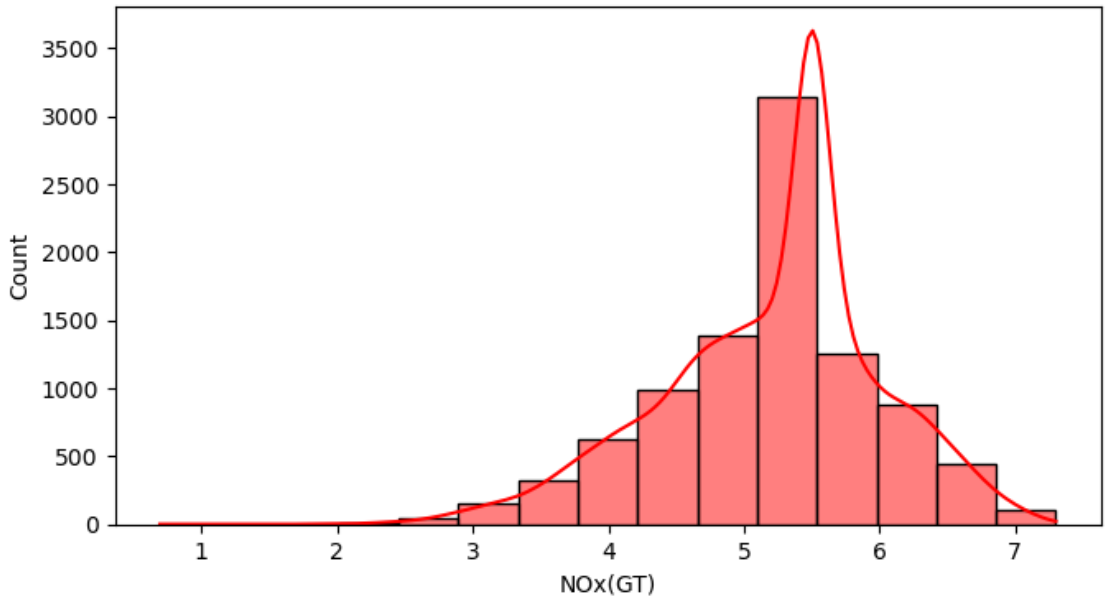
```
for i in analys_df_log.columns:
    if "GT" in i:
        plt.figure(figsize=(17,9))
        plt.subplot(2,2,1)
        sns.
↪histplot(x=analys_df_log[i],stat="count",color="red",bins=15,kde={'alpha':0.
↪5})
```

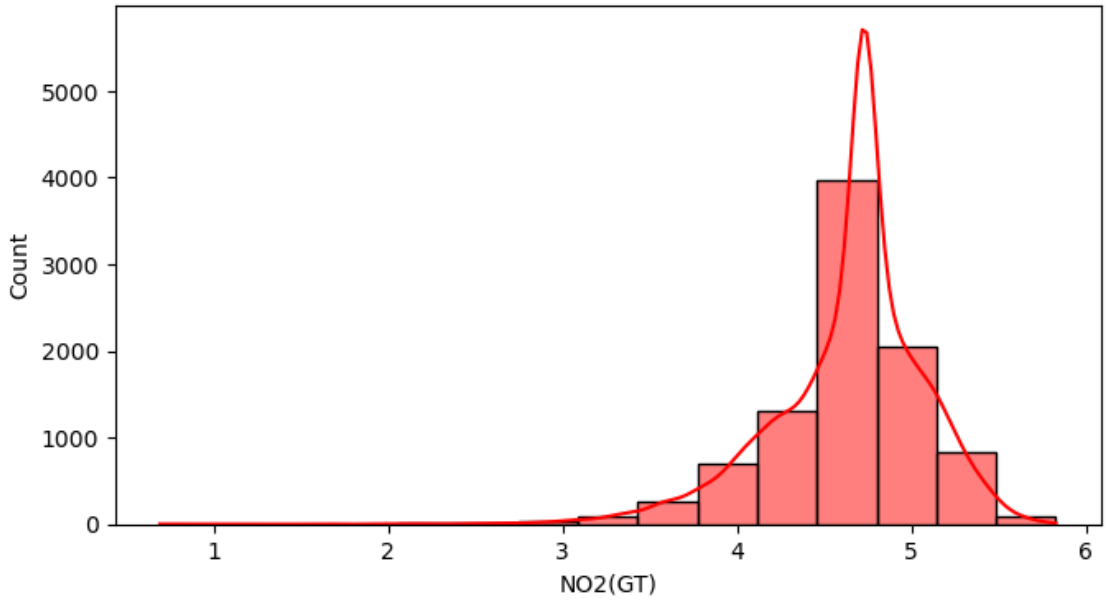
```
c:\Users\Mario\anaconda3\envs\bigdata_env\lib\site-
packages\pandas\core\arraylike.py:402: RuntimeWarning: divide by zero
encountered in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
c:\Users\Mario\anaconda3\envs\bigdata_env\lib\site-
packages\pandas\core\arraylike.py:402: RuntimeWarning: invalid value encountered
in log
  result = getattr(ufunc, method)(*inputs, **kwargs)
```




20000
15000
10000
5000
0
Count
2
3
4
5
6
7
NMHC(GT)

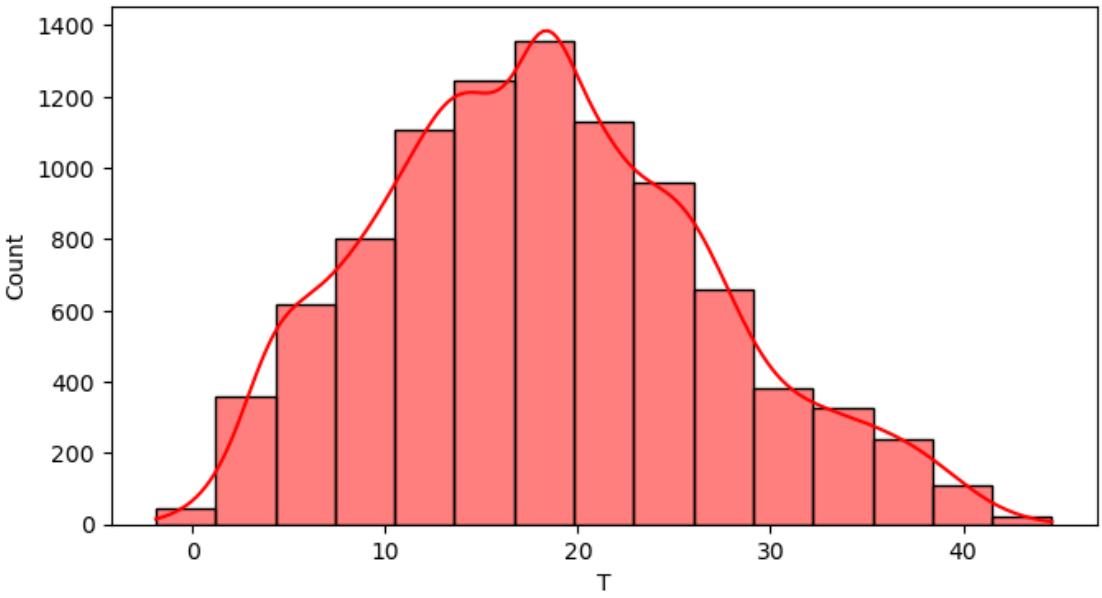

2000
1500
1000
500
0
Count
−2
−1
0
1
2
3
4
C6H6(GT)

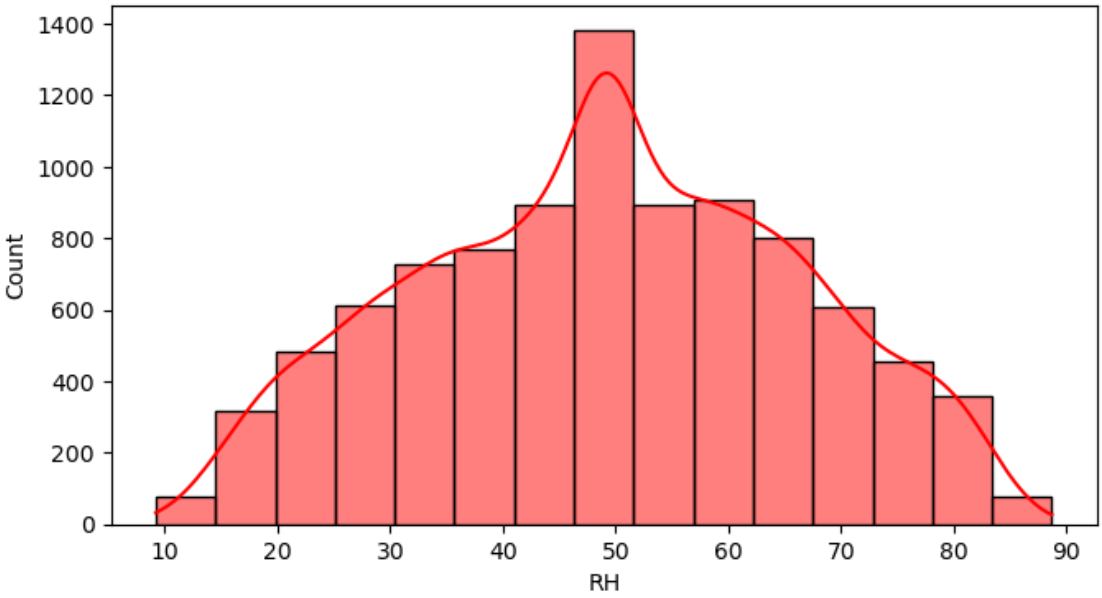


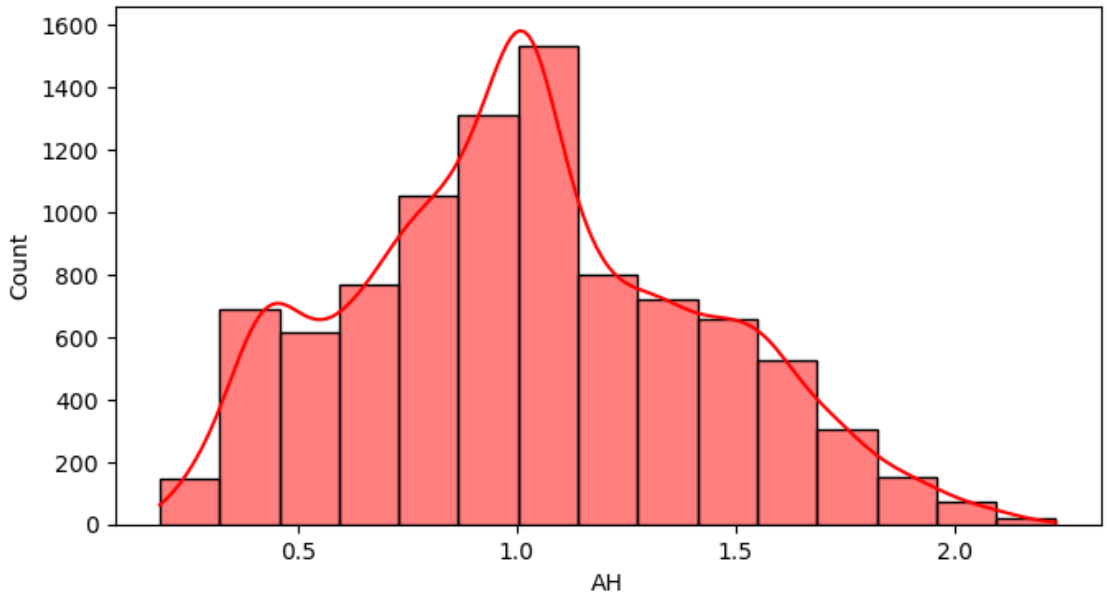


The graph above shows the result of logaritmic data from non sensors

[ ]:
```
for i in ['T', 'RH', 'AH']:
    plt.figure(figsize=(17,9))
    plt.subplot(2,2,1)
    sns.histplot(x=analys_df[i],stat="count",color="red",bins=15,kde={'alpha':0.
↪5})
```

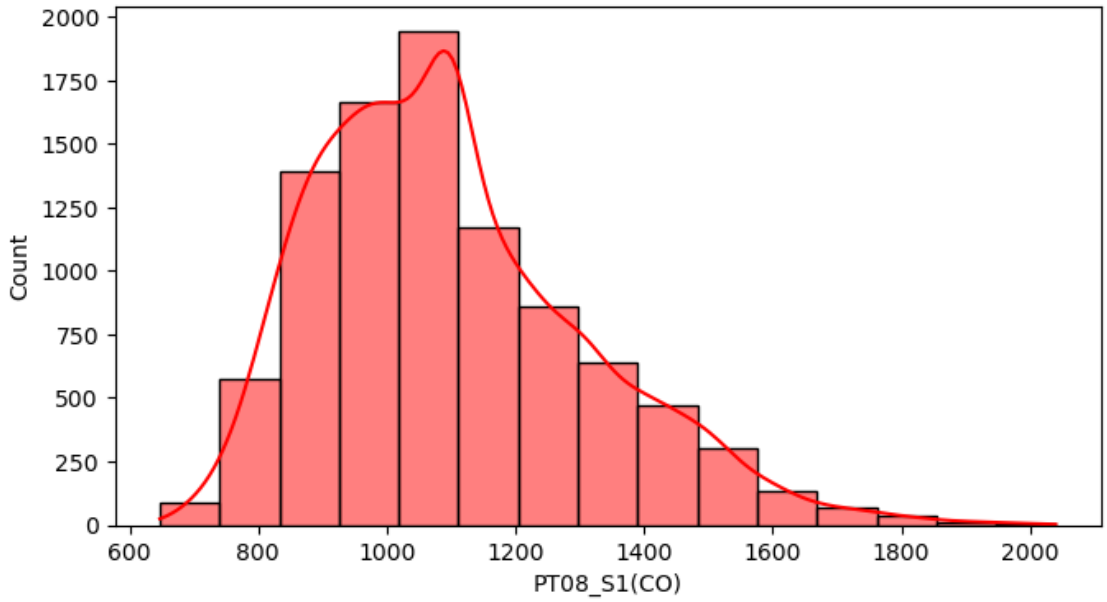

1400
1200
1000
800
600
400
200
0
Count
0
10
20
30
40
T

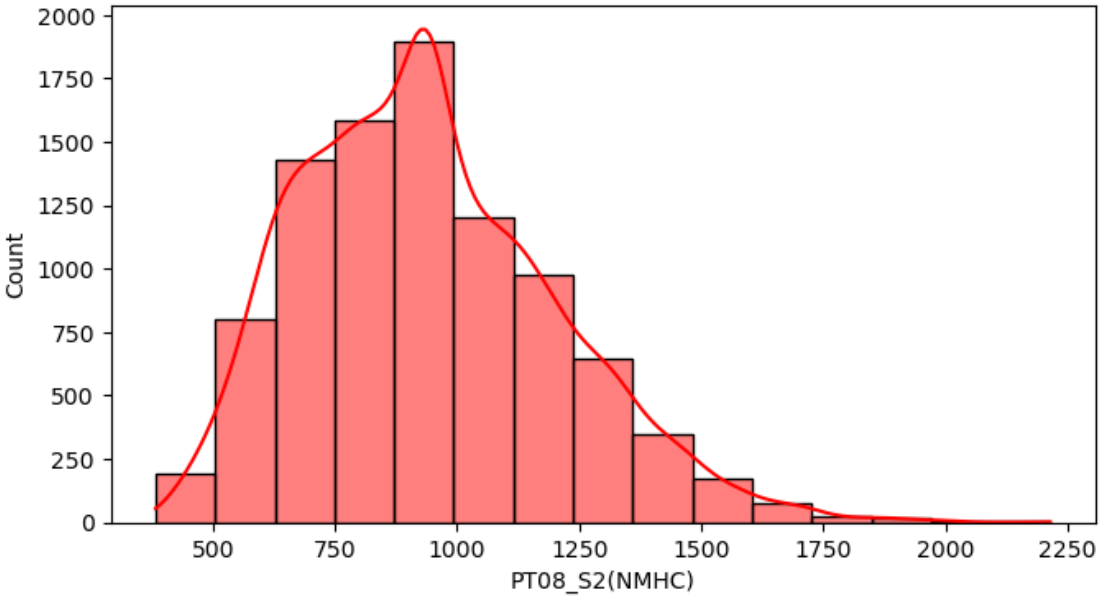

1400
1200
1000
800
600
400
200
0
Count
10
20
30
40
50
60
70
80
90
RH

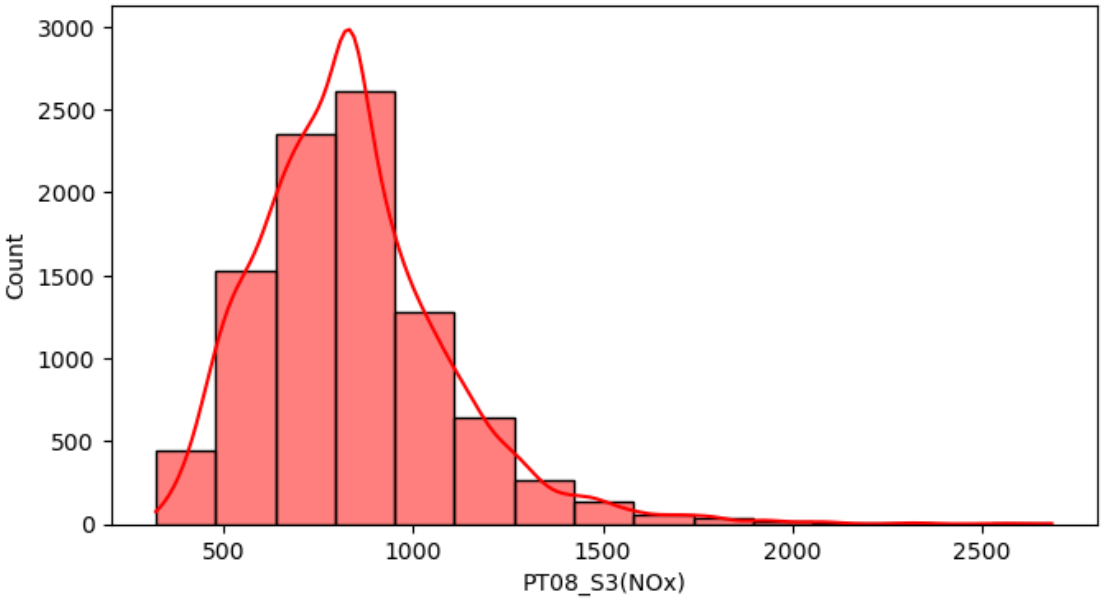


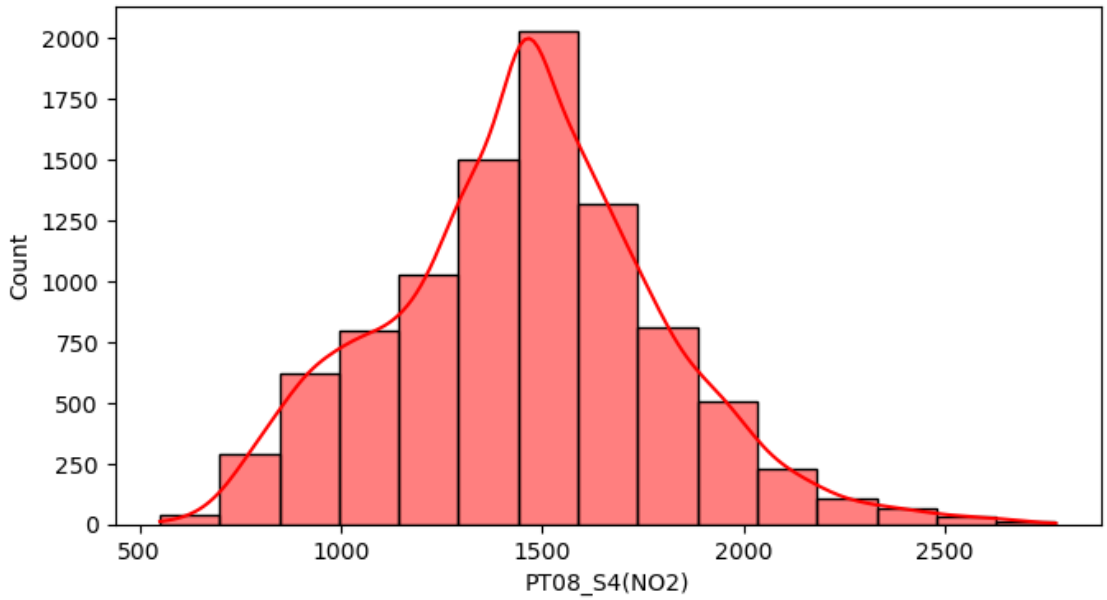
From the hisplot above we know that AH and T has left skewed data distribution. RH has normal distributed data

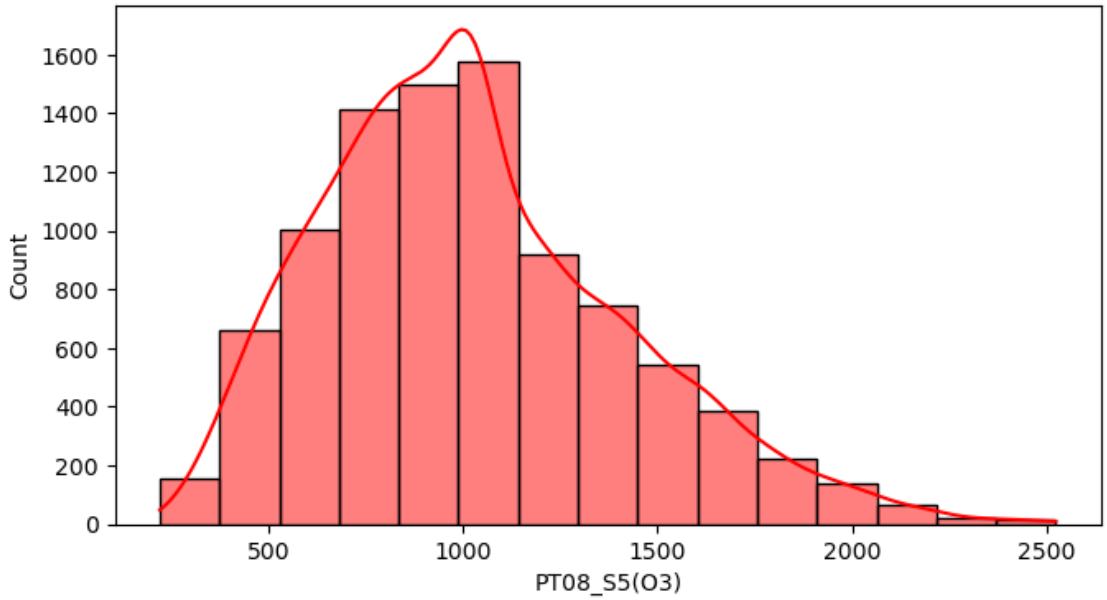
[ ]:
```
for i in analys_df.columns.drop(['Date','Time','datetime']):
    if "PT" in i:
        plt.figure(figsize=(17,9))
        plt.subplot(2,2,1)
        sns.
↪histplot(x=analys_df[i],stat="count",color="red",bins=15,kde={'alpha':0.5})
```




Count
PT08_S2(NMHC)


Count
PT08_S3(NOx)

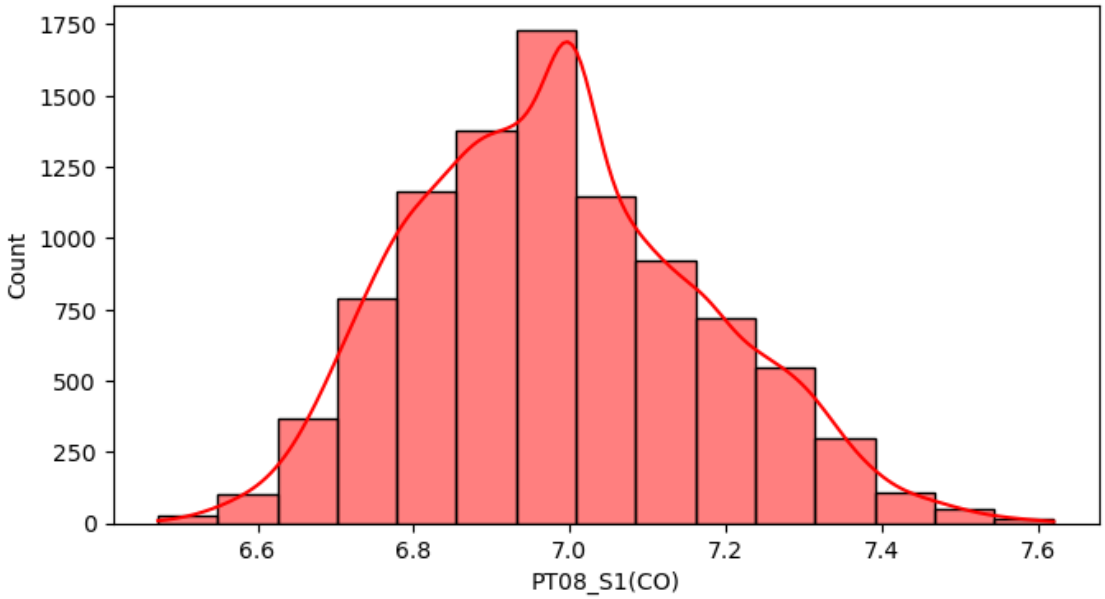


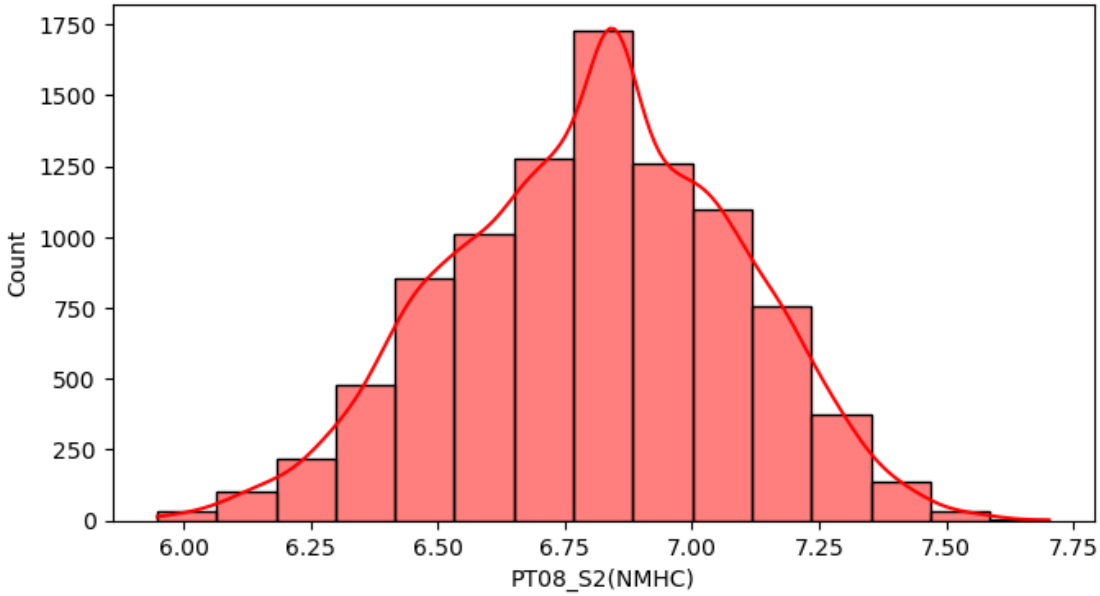


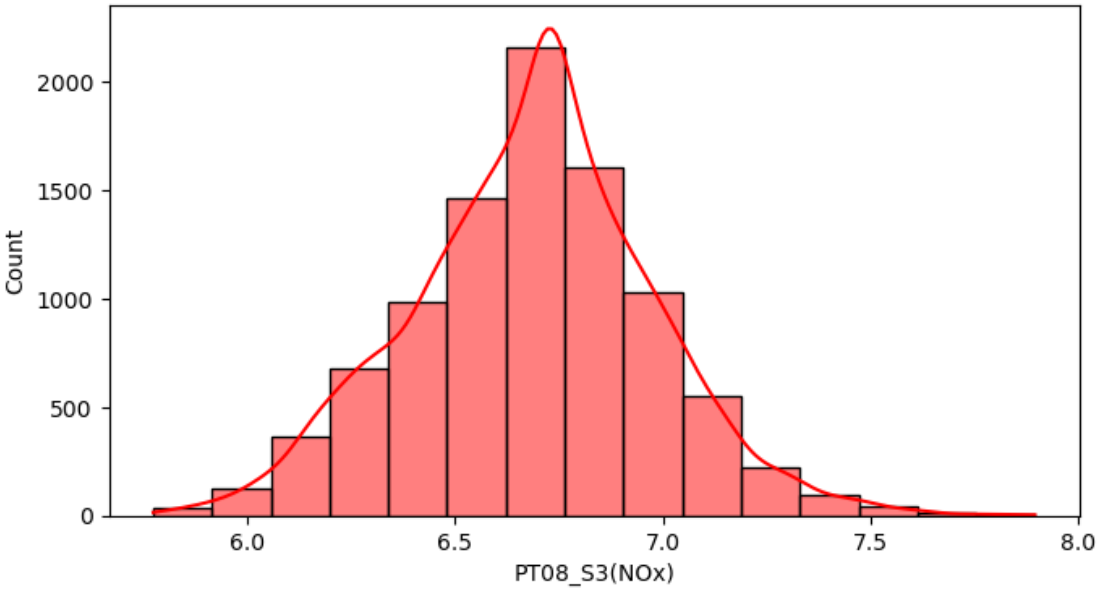
From the histplot above we know that for sensors device data only PT08_S4(NO2) has normaly distributed data

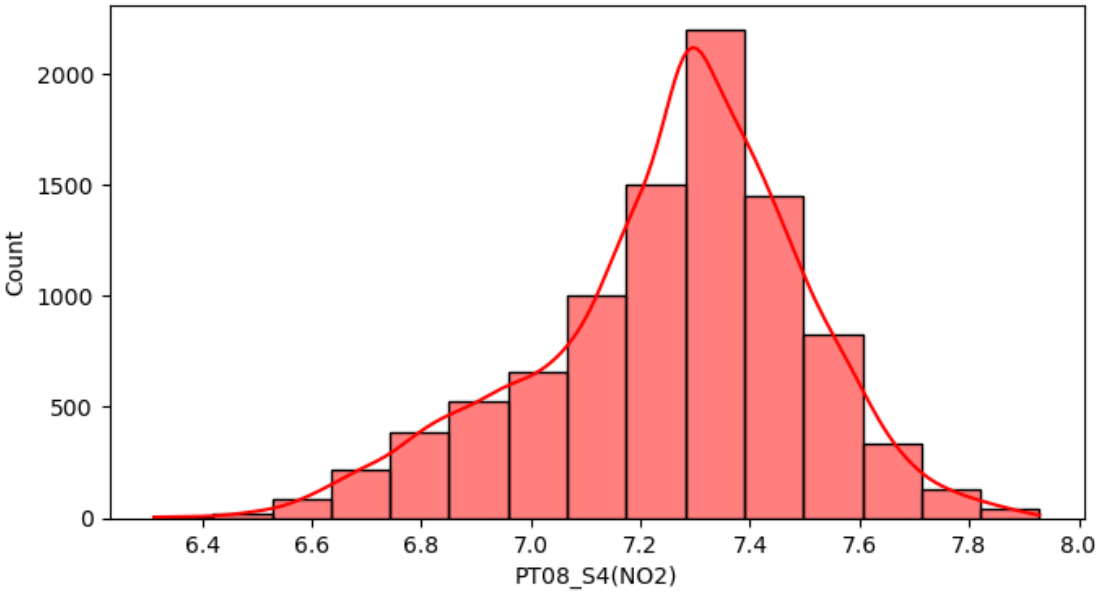
[ ]:
```
for i in analys_df_log.columns:
    if "PT" in i:
        plt.figure(figsize=(17,9))
        plt.subplot(2,2,1)
```

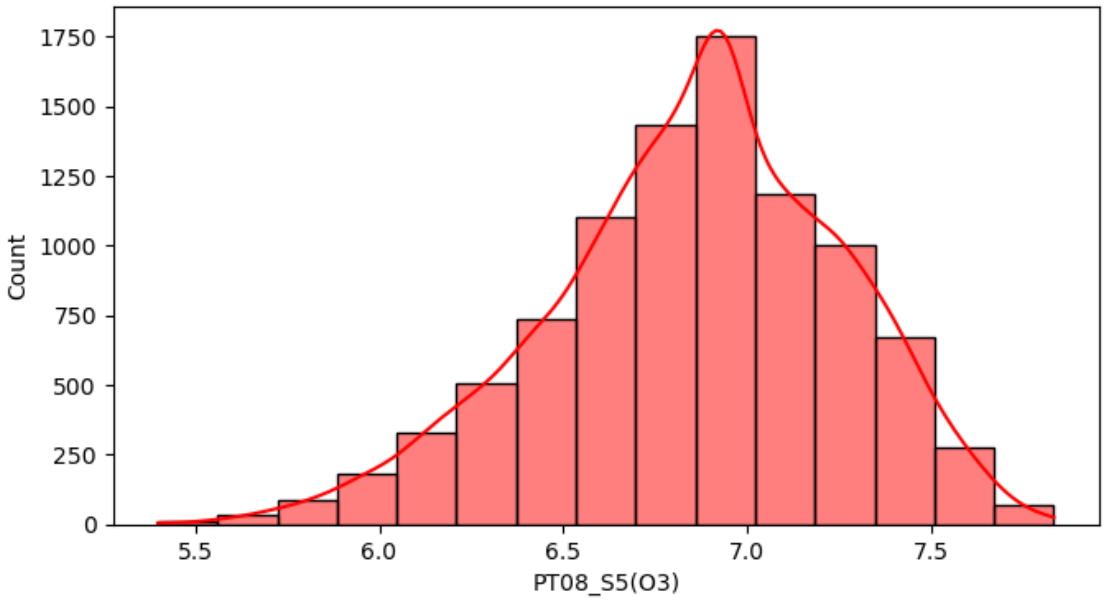
```
        sns.
↪histplot(x=analys_df_log[i],stat="count",color="red",bins=15,kde={'alpha':0.
↪5})
```






Count
2000
1500
1000
500
0
6.0
6.5
7.0
7.5
8.0
PT08_S3(NOx)


Count
2000
1500
1000
500
0
6.4
6.6
6.8
7.0
7.2
7.4
7.6
7.8
8.0
PT08_S4(NO2)



the graph above shows the logaritmic distribution data from sensors data

[ ]:
```
analys_df.skew()
```

```
C:\Users\Mario\AppData\Local\Temp\ipykernel_15636\2849351225.py:1:
FutureWarning: The default value of numeric_only in DataFrame.skew is
deprecated. In a future version, it will default to False. In addition,
specifying 'numeric_only=None' is deprecated. Select only valid columns or
specify the value of numeric_only to silence this warning.
  analys_df.skew()
```

[ ]:
```
CO(GT)           1.512462
PT08_S1(CO)      0.771593
NMHC(GT)         5.008751
C6H6(GT)         1.388959
PT08_S2(NMHC)    0.572947
NOx(GT)          1.891563
PT08_S3(NOx)     1.124152
NO2(GT)          0.685756
PT08_S4(NO2)     0.209617
PT08_S5(O3)      0.640784
T                0.315588
RH              -0.038692
AH               0.256452
dtype: float64
```

[ ]:
```
analys_df_recip = analys_df[analys_df.columns.drop(['Date','Time','datetime'])].
↪apply(lambda x: 1/x)

analys_df_recip.skew()
```

[ ]:
```
CO(GT)            6.202992
PT08_S1(CO)       0.159924
NMHC(GT)         12.064475
C6H6(GT)         11.218489
PT08_S2(NMHC)     0.848054
NOx(GT)          13.742315
PT08_S3(NOx)      0.917047
NO2(GT)          22.574983
PT08_S4(NO2)      1.249215
PT08_S5(O3)       1.616393
T                      NaN
RH                2.123789
AH                2.022020
dtype: float64
```

[ ]:
```
analys_df_log.skew().sort_values(ascending=True)
```

[ ]:
```
NMHC(GT)        -3.573356
NO2(GT)         -1.120840
RH              -0.875294
CO(GT)          -0.769585
AH              -0.720641
C6H6(GT)        -0.708299
NOx(GT)         -0.523995
PT08_S4(NO2)    -0.513243
PT08_S5(O3)     -0.363414
PT08_S2(NMHC)   -0.126332
PT08_S3(NOx)     0.008279
PT08_S1(CO)      0.293085
T                     NaN
dtype: float64
```

[ ]:
```
result = {}

data_recip = analys_df_recip.skew()

data_log = analys_df_log.skew()

data_original = analys_df.drop(['Date','Time','datetime'],axis=1).skew()

for column in data_recip.index:
    value_recip = data_recip[column]
```

```
    value_log = data_log[column]
    value_original = data_original[column]

    min_value = min(abs(value_recip), abs(value_log), abs(value_original))

    if abs(value_recip) == min_value:
        result[column] = (value_recip, 'Reciprocal')
    elif abs(value_log) == min_value:
        result[column] = (value_log, 'Log')
    else:
        result[column] = (value_original, 'Original')

for column, (value, transform) in result.items():
    print(f'{column} ==> {value} ==> {transform}')
```

```
CO(GT) ==> -0.769584858170462 ==> Log
PT08_S1(CO) ==> 0.15992351605726565 ==> Reciprocal
NMHC(GT) ==> -3.5733557686337885 ==> Log
C6H6(GT) ==> -0.7082989847217865 ==> Log
PT08_S2(NMHC) ==> -0.1263324483352556 ==> Log
NOx(GT) ==> -0.5239951029625104 ==> Log
PT08_S3(NOx) ==> 0.008278588018567765 ==> Log
NO2(GT) ==> 0.6857556195692693 ==> Original
PT08_S4(NO2) ==> 0.2096173766279425 ==> Original
PT08_S5(O3) ==> -0.36341365031270195 ==> Log
T ==> 0.3155884763806343 ==> Original
RH ==> -0.03869203186717812 ==> Original
AH ==> 0.25645171136696526 ==> Original
```

[ ]:
```
column_transform = {}

for column, (value, transform) in result.items():
    column_transform[column] = transform

column_transform
```

[ ]:
```
{'CO(GT)': 'Log',
 'PT08_S1(CO)': 'Reciprocal',
 'NMHC(GT)': 'Log',
 'C6H6(GT)': 'Log',
 'PT08_S2(NMHC)': 'Log',
 'NOx(GT)': 'Log',
 'PT08_S3(NOx)': 'Log',
 'NO2(GT)': 'Original',
 'PT08_S4(NO2)': 'Original',
 'PT08_S5(O3)': 'Log',
 'T': 'Original',
 'RH': 'Original',
```

```
 'AH': 'Original'}
```

[ ]:
```
from scipy.stats import shapiro

for i in analys_df.columns.drop(['Date','Time','datetime']):
    p_vals = shapiro(analys_df[i])[1]
    sig_vals = shapiro(analys_df[i])[0]
    if p_vals < 0.05:
        print(f"[{i}] p-value: {p_vals}, signif-value {sig_vals}")
```

```
[CO(GT)] p-value: 0.0, signif-value 0.8922240734100342
[PT08_S1(CO)] p-value: 3.783505853677006e-44, signif-value 0.9615641236305237
[NMHC(GT)] p-value: 0.0, signif-value 0.30608034133911133
[C6H6(GT)] p-value: 0.0, signif-value 0.8967937231063843
[PT08_S2(NMHC)] p-value: 1.1544068402564529e-35, signif-value 0.9778857231140137
[NOx(GT)] p-value: 0.0, signif-value 0.8305597305297852
[PT08_S3(NOx)] p-value: 0.0, signif-value 0.9442952871322632
[NO2(GT)] p-value: 1.6675451725465323e-43, signif-value 0.9630635976791382
[PT08_S4(NO2)] p-value: 1.531480172573161e-21, signif-value 0.9926659464836121
[PT08_S5(O3)] p-value: 2.5989980708724885e-39, signif-value 0.9717391729354858
[T] p-value: 6.149836164993573e-28, signif-value 0.9874593615531921
[RH] p-value: 3.2198558663915922e-28, signif-value 0.9871805906295776
[AH] p-value: 4.135695920814417e-29, signif-value 0.9862651228904724

c:\Users\Mario\anaconda3\envs\bigdata_env\lib\site-
packages\scipy\stats\_morestats.py:1816: UserWarning: p-value may not be
accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

This process help us to choose what is the best method for handling the skewed data. Every column will calculated to determine what is the optimze distribution method that will be used for model development that in the end will be compared model with data transformed and the raw data.

4 Model Development

[ ]:
```
MODEL_MODULE_FILE = "module/airpolution_model_v1.py"
```

[ ]:
```
%%writefile {MODEL_MODULE_FILE}

import airpolution_data_constant
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.evaluation import RegressionEvaluator

_FEATURE_KEY = airpolution_data_constant.FEATURE_KEY
_LABEL_KEY = airpolution_data_constant.LABEL_KEY
_ALL_KEY = airpolution_data_constant.ALL_KEY
FEATURE_KEY_TN = [
    'CO(GT)_tn',
```

```
    'PT08_S1(CO)_tn',
    'NMHC(GT)_tn',
    'C6H6(GT)_tn',
    'PT08_S2(NMHC)_tn',
    'NOx(GT)_tn',
    'PT08_S3(NOx)_tn',
    'NO2(GT)_tn',
    'PT08_S4(NO2)_tn',
    'PT08_S5(O3)_tn',
    'RH_tn',
    'AH_tn']


def transform_data(input, assembler):
    """Vectorizing the data"""

    print("Vectorizing the data")

    transformed = assembler.transform(input)
    return transformed

def splitting_data(input):
    """Splitting the data"""

    print("Splitting the data")
    (train_data, test_data) = input.randomSplit([0.8, 0.2])

    return (train_data, test_data)

def train_model(model, input):
    """Inputing model from user"""

    trained_model = model.fit(input)
    return trained_model

def eval_model(pred_model):
    """evaluate model"""

    rmse = RegressionEvaluator(labelCol=_LABEL_KEY, predictionCol="prediction",␣
 ↪metricName="rmse")
    rmse = rmse.evaluate(pred_model)
    mae = RegressionEvaluator(labelCol=_LABEL_KEY, predictionCol="prediction",␣
 ↪metricName="mae")
    mae = mae.evaluate(pred_model)
    r2 = RegressionEvaluator(labelCol=_LABEL_KEY, predictionCol="prediction",␣
 ↪metricName="r2")
    r2 = r2.evaluate(pred_model)
```

```
    print("RMSE: ", rmse)
    print("MAE: ", mae)
    print("R-squared: ", r2)

def model_fn(model_list, input, assembler_transform):

    if assembler_transform:
            assembler = VectorAssembler(inputCols=FEATURE_KEY_TN,␣
 ↪outputCol="features")
    else:
            assembler = VectorAssembler(inputCols=_FEATURE_KEY,␣
 ↪outputCol="features")

    transformed_data = transform_data(input, assembler)
    train_data, test_data = splitting_data(transformed_data)

    for model in model_list:
        print("\n,", model)
        trained_model = train_model(model, train_data)
        pred_model = trained_model.transform(test_data)
        model_evaluation = eval_model(pred_model)
        print("\n")

    return trained_model
```

```
Overwriting module/airpolution_model_v1.py
```

[ ]:
```
import airpolution_model_v1
from pyspark.ml.regression import DecisionTreeRegressor, RandomForestRegressor

model_list = [
    DecisionTreeRegressor(labelCol="T", featuresCol="features"),
    RandomForestRegressor(labelCol="T", featuresCol="features")
]

df_model = airpolution_model_v1.model_fn(model_list, df,␣
 ↪assembler_transform=False)
```

```
Vectorizing the data
Splitting the data

, DecisionTreeRegressor_4c8a8baa14c8
RMSE:  2.873588335180391
MAE:  2.241733756743419
R-squared:  0.9956220651287081
```

```
, RandomForestRegressor_c1bb308f0674
RMSE:  3.524948499588271
MAE:  2.6944294337508787
R-squared:  0.9934124223898312
```

4.1 Model Development for distribution transformed data

[ ]:
```
df_processing_dist = airpolution_data_transform.preprocessing_fn(df,␣
↪dist_transform=True)
```

[ ]:
```
df_processing_dist.show()
```

```
+----------+--------+--------------------+--------------------+-----------------
-+------------------+-----------------+-----------------+------------------+--
-----+------------+-----------------+----+----+------+-------------------+
|      Date|    Time|              CO(GT)|         PT08_S1(CO)|
NMHC(GT)|           C6H6(GT)|     PT08_S2(NMHC)|            NOx(GT)|
PT08_S3(NOx)|NO2(GT)|PT08_S4(NO2)|       PT08_S5(O3)|   T|  RH|    AH|
datetime|
+----------+--------+--------------------+--------------------+-----------------
-+------------------+-----------------+-----------------+------------------+--
-----+------------+-----------------+----+----+------+-------------------+
|2004-03-11|07:00:00| 0.09531017980432493|8.741258741258741E-4|
3.367295829986474|1.1631508098056809| 6.502790045915623| 4.584967478670572|
7.306531398939505|     82|        1339|
6.593044534142437|10.2|59.6|0.7417|2004-03-11 07:00:00|
|2004-03-14|14:00:00|  0.5877866649021191|8.285004142502071E-4|
4.430816798843313|2.0149030205422647| 6.778784897685177| 4.634728988229636|
7.00669522683704|    102|        1490|
6.77078942390898|21.4|30.2|0.7616|2004-03-14 14:00:00|
|2004-03-15|09:00:00|  2.0918640616783932|5.099439061703213E-4|
6.42648845745769|3.6027767550605247| 7.438971592395862| 6.169610732491456|
6.285998094508865|    149|        2665|
7.688913336864796|14.8|54.3|0.9076|2004-03-15 09:00:00|
|2004-03-19|22:00:00|  0.7884573603642703| 8.51063829787234E-4|
5.384495062789089|2.2082744135228043| 6.851184927493743| 4.962844630259907|
6.806829360392176|    116|        1604|
6.985641817639208|14.7|57.6|0.9573|2004-03-19 22:00:00|
|2004-03-22|17:00:00|  0.9555114450274365|8.680555555555555E-4|
5.220355825078325| 2.517696472610991| 6.967909201801884| 4.927253685157205|
6.833031732786201|    103|        1606|
6.745236349484362|20.2|28.5|0.6682|2004-03-22 17:00:00|
|2004-03-24|22:00:00|  0.5306282510621704|9.551098376313276E-4|
4.574710978503383|1.7749523509116738| 6.695798917058491| 4.770684624465665|
6.96979066990159|    108|        1435|
```

```
6.641182169740591|10.1|66.9|0.8248|2004-03-24 22:00:00|
|2004-04-08|04:00:00|
0.7667458827323177|0.0012150668286755 77|3.6375861597263857|0.5877866649021191|
6.342121418721152|3.7612001156935624|7.2196420401307355|     57|        1263|
6.54965074223381| 8.3|75.6|0.8302|2004-04-08 04:00:00|
|2004-04-29|01:00:00|                 0.0|9.596928982725527E-4|
4.430816798843313|1.6486586255873816| 6.648984550024776|3.9318256327243257|
6.827629234502852|     61|        1495|
6.875232087276577|16.8|59.5|1.1269|2004-04-29 01:00:00|
|2004-04-29|08:00:00|  1.9740810260220096| 5.64652738565782E-4|
7.029087564149662|3.5890591188317256| 7.432483807917119| 5.908082938168931|
6.133398042996649|    125|        2572|
7.591861714889934|16.1|56.5|1.0274|2004-04-29 08:00:00|
|2004-05-03|03:00:00|  -0.916290731874155|0.001146788990825…|
5.384495062789089|0.5306282510621704| 6.335054251498059|5.5053315359323625|
7.157735484249907|    113|        1385|
6.202535517187923|15.6|65.3|1.1484|2004-05-03 03:00:00|
|2004-05-06|10:00:00|  0.9162907318741551|9.587727708533077E-4|
5.384495062789089|2.5257286443082556|  6.97166860472579| 5.081404364984463|
6.670766320845874|    114|        1692|
6.921658184151129|20.2|37.7|0.8833|2004-05-06 10:00:00|
|2004-05-09|17:00:00|  0.7667458827323177|0.001033057851239…|
5.384495062789089|1.6863989535702288| 6.665683717782408|5.5053315359323625|
6.918695219020472|    113|        1446|
6.309918278226516|17.8|44.4|0.8967|2004-05-09 17:00:00|
|2004-05-15|02:00:00|                 0.0|0.001069518716577…|
5.384495062789089|1.6863989535702288| 6.666956792429207| 4.007333185232471|
6.871091294610546|     71|        1389|
6.860663671448287|17.4|43.4|0.8553|2004-05-15 02:00:00|
|2004-06-05|12:00:00| 0.33647223662121284|0.001094091903719…|
5.384495062789089|2.0794415416798357| 6.802394763324311| 4.454347296253507|
6.917705609835305|     68|        1596|
6.559615237493242|30.7|26.0|1.1287|2004-06-05 12:00:00|
|2004-06-07|04:00:00|  0.7667458827323177|0.001377410468319…|
5.384495062789089|0.7419373447293773|6.3818160174060985| 3.091042453358316|
7.141245122350491|     34|        1386| 6.361302477572996|17.2|57.9|
1.127|2004-06-07 04:00:00|
|2004-06-21|13:00:00|  0.1823215567939546|0.001193317422434…|
5.384495062789089|1.9740810260220096|6.7661917146603505| 4.189654742026425|
6.912742820493176|     76|        1454|
6.29156913955832|30.1|18.5|0.7747|2004-06-21 13:00:00|
|2004-06-29|17:00:00|   1.308332819650179|7.593014426727411E-4|
5.384495062789089| 3.068052935133617| 7.198931240688173| 5.298317366548036|
6.385194398997726|    198|        2132|
7.216709486709457|37.1|26.9|1.6635|2004-06-29 17:00:00|
|2004-07-04|13:00:00|-0.35667494393873245|0.001131221719457…|
5.384495062789089|1.3609765531356006| 6.555356891810665|3.5263605246161616|
6.98100574072173|     46|        1394|6.1903154058531475|37.9|18.8|
```

```
1.222|2004-07-04 13:00:00|
|2004-07-08|02:00:00| -0.5108256237659907|0.001039501039501…|
5.384495062789089|1.7578579175523736| 6.688354713946762|3.7612001156935624|
6.748759547491679|     60|        1507|  6.97166860472579|27.5|34.0|
1.231|2004-07-08 02:00:00|
|2004-07-12|05:00:00| -0.6931471805599453|0.001197604790419…|
5.384495062789089|1.1939224684724346|  6.510258340523 15|3.7612001156935624|
6.959398512133975|     43|
1423|6.4425401664681985|20.0|51.8|1.1967|2004-07-12 05:00:00|
+----------+--------+--------------------+--------------------+-----------------
-+------------------+-----------------+------------------+------------------+--
-----+-----------+-----------------+----+----+------+------------------+
only showing top 20 rows
```

[ ]:
```
import airpolution_model_v1
from pyspark.ml.regression import DecisionTreeRegressor, RandomForestRegressor

model_list = [
    DecisionTreeRegressor(labelCol="T", featuresCol="features"),
    RandomForestRegressor(labelCol="T", featuresCol="features")
]

df_model_transform = airpolution_model_v1.model_fn(model_list,␣
 ↪df_processing_dist, assembler_transform=False)
```

```
Vectorizing the data
Splitting the data

, DecisionTreeRegressor_05fb392ef537
RMSE:  2.1010514893241146
MAE:  1.7132328864716941
R-squared:  0.9415494509096733



, RandomForestRegressor_975aad9cd693
RMSE:  2.8845030172402546
MAE:  2.257871719710507
R-squared:  0.8898315601129945
```

4.2 Model Development for distribution normalize data

[ ]:
```
df_processing_normalize = airpolution_data_transform.preprocessing_fn(df,␣
↪normalize=True)
```

[ ]:
```
df_processing_normalize.show()
```

```
+----------+--------+-----------------+----------+--------+--------+----------
---+-------+-----------+-------+-----------+----------+----+----+------+-----
-------------+--------------------+--------------------+--------------------+--
-----------------+--------------------+--------------------+------------------
-+--------------------+--------------------+--------------------+--------------
-----+--------------------+--------------------+
|      Date|    Time|            CO(GT)|PT08_S1(CO)|NMHC(GT)|C6H6(GT)|PT08_S2(NM
HC)|NOx(GT)|PT08_S3(NOx)|NO2(GT)|PT08_S4(NO2)|PT08_S5(O3)|   T|  RH|    AH|
datetime|           CO(GT)_tn|      PT08_S1(CO)_tn|         NMHC(GT)_tn|
C6H6(GT)_tn|    PT08_S2(NMHC)_tn|          NOx(GT)_tn|     PT08_S3(NOx)_tn|
NO2(GT)_tn|     PT08_S4(NO2)_tn|      PT08_S5(O3)_tn|               T_tn|
RH_tn|              AH_tn|
+----------+--------+-----------------+----------+--------+--------+----------
---+-------+-----------+-------+-----------+----------+----+----+------+-----
-------------+--------------------+--------------------+--------------------+--
-----------------+--------------------+--------------------+------------------
-+--------------------+--------------------+--------------------+--------------
-----+--------------------+--------------------+
|2004-03-11|07:00:00|              1.1|       1144|      29|     3.2|
667|     98|        1490|     82|        1339|
730|10.2|59.6|0.7417|2004-03-11 07:00:00|[0.0847457627118644]|[0.35678391959798.
..|[0.01861252115059…|[0.04874213836477…|[0.15510649918077…|[0.06499661475
964…|[0.49470563320626…|[0.23668639053254…|[0.35431654676258…|[0.2211120
764552563]|[0.2602150537634409]|[0.6339622641509435]|[0.27219860235547…|
|2004-03-14|14:00:00|              1.8|       1207|      84|     7.5|
879|    103|        1104|    102|        1490|
872|21.4|30.2|0.7616|2004-03-14 14:00:00|[0.14406779661016…|[0.402010050251256
3]|[0.06514382402707…|[0.11635220125786…|[0.2708902239213544]|[0.06838185511
171…|[0.3312155866158407]|[0.2958579881656805]|[0.4222122302158274]|[0.2827975
6733275…|[0.5010752688172043]|[0.2641509433962264]|[0.2819234716317256]|
|2004-03-15|09:00:00|              8.1|       1961|     618|    36.7|
1701|    478|         537|    149|        2665|
2184|14.8|54.3|0.9076|2004-03-15 09:00:00|[0.6779661016949152]|[0.94328786791098
35]|[0.5169204737732657]|[0.5754716981132075]|[0.7198252321135991]|[0.3222748815
165877]|[0.09106310885218…|[0.4349112426035503]|[0.9505395683453238]|[0.852736
7506516073]|[0.35913978494623…|[0.5672955974842767]|[0.35327175878414…|
|2004-03-19|22:00:00|              2.2|       1175|     218|     9.1|
945|    143|         904|    116|        1604|
1081|14.7|57.6|0.9573|2004-03-19 22:00:00|[0.17796610169491…|[0.37903804737975
…|[0.1785109983079526]|[0.14150943396226…|[0.30693610049153…|[0.0954637779
282329]|[0.2465057179161372]|[0.33727810650887…|[0.47347122302158…|[0.373588
```

```
1841876629]|[0.35698924731182…| [0.608805031446541]|[0.37755949762986…|
|2004-03-22|17:00:00|                2.6|       1152|     185|    12.4|
1062|    138|         928|    103|        1606|
850|20.2|28.5|0.6682|2004-03-22 17:00:00| [0.211864406779661]|[0.36252692031586.
..|[0.1505922165820643]|[0.19339622641509…|[0.37083560895685…|[0.09207853757
616…|[0.25667090216010…|[0.29881656804733…|[0.47437050359712…|[0.2732406
602953953]|[0.4752688172043011]|[0.2427672955974843]| [0.236280115330108]|
|2004-03-24|22:00:00|                1.7|       1047|      97|     5.9|
809|    118|        1064|    108|        1435|
766|10.1|66.9|0.8248|2004-03-24 22:00:00|[0.13559322033898…|[0.287150035893754
5]|[0.07614213197969…|[0.09119496855345…|[0.23265974877116…|[0.07853757616
790…|[0.31427361287590…|[0.3136094674556213]|[0.39748201438848…| [0.236750
651607298]|[0.25806451612903…|[0.7257861635220126]|[0.31280848360455…|
|2004-04-08|04:00:00|2.1527495439145157|        823|      38|     1.8|
568|     43|        1366|     57|        1263|        699|
8.3|75.6|0.8302|2004-04-08 04:00:00|[0.17396182575546…|[0.12634601579325…|[0
.02622673434856…|[0.02672955974842…|[0.10103768432550…|[0.02775897088693..
.|[0.4421855146124523]|[0.16272189349112…|[0.32014388489208…|[0.207645525629
88…|[0.21935483870967…|[0.8352201257861634]|[0.3154473928553976]|
|2004-04-29|01:00:00|                1.0|       1042|      84|     5.2|
772|     51|         923|     61|        1495|
968|16.8|59.5|1.1269|2004-04-29 01:00:00|[0.07627118644067…|[0.28356066044508.
..|[0.06514382402707…|[0.08018867924528…|[0.21245221190606…|[0.03317535545
023…|[0.25455315544260…|[0.17455621301775…|[0.4244604316546763]|[0.3245004
344048653]|[0.4021505376344086]|[0.6327044025157232]|[0.4604407955822704]|
|2004-04-29|08:00:00|                7.2|       1771|    1129|    36.2|
1690|    368|         461|    125|        2572|
1982|16.1|56.5|1.0274|2004-04-29
08:00:00|[0.6016949152542374]|[0.8068916008614502]|
[0.949238578680203]|[0.5676100628930818]| [0.713817586018569]|[0.24779959377115.
..|[0.05887335874629…|[0.36390532544378…|[0.9087230215827339]|  [0.764986967
85404]|[0.38709677419354…|[0.5949685534591195]|[0.41181644920099…|
|2004-05-03|03:00:00|                0.4|        872|     218|     1.7|
564|    246|        1284|    113|        1385|
494|15.6|65.3|1.1484|2004-05-03 03:00:00|[0.02542372881355…|[0.161521895190236
9]|[0.1785109983079526]|[0.02515723270440…|[0.09885308574549…|[0.16519972918
077…|[0.40745446844557…|[0.32840236686390…|
[0.375]|[0.11859252823631…|[0.37634408602150…|[0.7056603773584905]|
[0.470947563895812]|
|2004-05-06|10:00:00|                2.5|       1043|     218|    12.5|
1066|    161|         789|    114|        1692|
1014|20.2|37.7|0.8833|2004-05-06 10:00:00|[0.20338983050847…|[0.28427853553481
…|[0.1785109983079526]|[0.1949685534591195]|[0.37302020753686…|[0.1076506431
9566…|[0.1977975434138077]|[0.3313609467455621]|[0.5130395683453237]|[0.344483
05821025…|[0.4752688172043011]|[0.35849056603773…|[0.34139666715535…|
|2004-05-09|17:00:00|2.1527495439145157|        968|     218|     5.4|
785|    246|        1011|    113|        1446|
550|17.8|44.4|0.8967|2004-05-09 17:00:00|[0.17396182575546…|[0.23043790380473.
```

```
..|[0.1785109983079526]|[0.08333333333333…|[0.21955215729109…|[0.16519972918
077…|[0.2918254976704786]|[0.32840236686390…|[0.40242805755395…|[0.1429192
0069504…|[0.42365591397849…|[0.44276729559748…|[0.3479450715926306]|
|2004-05-15|02:00:00|                1.0|         935|     218|     5.4|
786|     55|          964|     71|         1389|
954|17.4|43.4|0.8553|2004-05-15 02:00:00|[0.07627118644067…|[0.20674802584350.
..|[0.1785109983079526]|[0.08333333333333…|[0.22009830693610…|[0.03588354773
188…|[0.2719186785260483]|[0.20414201183431…|[0.3767985611510792]|[0.3184187
6629018…|[0.4150537634408602]|[0.4301886792452831]|[0.32771343400283…|
|2004-06-05|12:00:00|                1.4|         914|     218|     8.0|
900|     86|         1010|     68|         1596|
706|30.7|26.0|1.1287|2004-06-05 12:00:00|[0.11016949152542…|[0.19167264895908.
..|[0.1785109983079526]|[0.12421383647798…|[0.2823593664664118]|[0.05687203791
469…|[0.2914019483269801]|[0.1952662721893491]|[0.4698741007194245]|[0.2106863
5968722…|[0.7010752688172044]|[0.21132075471698…|[0.46132043199921…|
|2004-06-07|04:00:00|2.1527495439145157|         726|     218|     2.1|
591|     22|         1263|     34|         1386|         579|17.2|57.9|
1.127|2004-06-07 04:00:00|[0.17396182575546…|[0.05671213208901…|[0.178510998
3079526]|[0.03144654088050…|[0.11359912616056…|[0.01354096140825…| [0.3985
59932232105]|[0.09467455621301…|[0.3754496402877698]|[0.15551694178974…|
[0.410752688172043]|[0.6125786163522013]|[0.46048966427210…|
|2004-06-21|13:00:00|                1.2|         838|     218|     7.2|
868|     66|         1005|     76|         1454|
540|30.1|18.5|0.7747|2004-06-21 13:00:00|[0.09322033898305…|[0.137114142139267
8]|[0.1785109983079526]|[0.11163522012578…|[0.2648825778263244]|[0.04333107650
643…|[0.2892842016094875]|[0.21893491124260…|[0.40602517985611…|[0.1385751
5204170…|[0.6881720430107527]|[0.11698113207547…|[0.28832526999951…|
|2004-06-29|17:00:00|                3.7|        1317|     218|    21.5|
1338|    200|          593|    198|         2132|
1362|37.1|26.9|1.6635|2004-06-29 17:00:00|[0.3050847457627119]|[0.48097631012203
88]|[0.1785109983079526]|[0.33647798742138…|[0.5215729109776078]|[0.1340555179
4177…|[0.11478187208809…|[0.5798816568047337]|[0.7108812949640289]|[0.495655
9513466551]|[0.8387096774193549]|[0.22264150943396…|[0.7226701852123344]|
|2004-07-04|13:00:00|                0.7|         884|     218|     3.9|
703|     34|         1076|     46|         1394|         488|37.9|18.8|
1.222|2004-07-04 13:00:00|[0.05084745762711…|[0.17013639626704…|[0.178510998
3079526]|[0.05974842767295…|[0.17476788640087…|[0.02166553825321…|[0.31935
620499788…|[0.1301775147928994]|[0.3790467625899281]|[0.11598609904430…|[0.8
559139784946237]|[0.1207547169811321]|[0.5069149196110051]|
|2004-07-08|02:00:00|                0.6|         962|     218|     5.8|
803|     43|          853|     60|         1507|        1066|27.5|34.0|
1.231|2004-07-08 02:00:00|[0.0423728813559322]|[0.22613065326633…|[0.178510998
3079526]|[0.08962264150943…|[0.2293828509011469]|[0.02775897088693…|[0.22490
470139771…|[0.17159763313609…|[0.4298561151079137]|[0.3670721112076455]|[0.6
322580645161291]|[0.31194968553459…|[0.5113131016957435]|
|2004-07-12|05:00:00|                0.5|         835|     218|     3.3|
672|     43|         1053|     43|         1423|
628|20.0|51.8|1.1967|2004-07-12 05:00:00|[0.03389830508474…|[0.134960516870064
```

```
6]|[0.1785109983079526]|[0.05031446540880…|[0.15783724740578…|[0.02775897088
693…|[0.30961457009741…|[0.12130177514792…|[0.3920863309352518]|[0.1768027
8019113…|[0.47096774193548…|[0.5358490566037736]|[0.4945511410839075]|
+---------+-------+-----------------+----------+-------+-------+---------
---+-------+-----------+-------+-----------+----------+----+----+------+-----
-------------+--------------------+--------------------+--------------------+--
------------------+--------------------+--------------------+-----------------
-+--------------------+--------------------+--------------------+-------------
-----+--------------------+--------------------+
only showing top 20 rows
```

[ ]:
```
import airpolution_model_v1
from pyspark.ml.regression import DecisionTreeRegressor, RandomForestRegressor

model_list = [
    DecisionTreeRegressor(labelCol="T", featuresCol="features"),
    RandomForestRegressor(labelCol="T", featuresCol="features")
]

df_model = airpolution_model_v1.model_fn(model_list, df_processing_normalize,␣
 ↪assembler_transform=True)
```

```
Vectorizing the data
Splitting the data

, DecisionTreeRegressor_92d80f657443
RMSE:  2.044640645991832
MAE:  1.6443691591966
R-squared:  0.9455756805363574



, RandomForestRegressor_075e5636fb6d
RMSE:  2.7702104107851593
MAE:  2.1196213694982124
R-squared:  0.9000956171506417
```

4.3 Model Development for log transform and normalize data

[ ]:
```
df_processing_dist_norm = airpolution_data_transform.preprocessing_fn(df,␣
 ↪dist_transform=True, normalize=True)
```

[ ]:
```
df_processing_dist_norm.show()
```

```
+---------+-------+-----------------+-------------------+-----------------
-+-----------------+-----------------+-----------------+-----------------+--
```

```
-----+-----------+-----------------+----+----+------+------------------+-----
--------------+-------------------+-------------------+-------------------+-
-----------------+-------------------+-------------------+-----------------
--+-------------------+-------------------+-------------------+-------------
------+-------------------+
|      Date|    Time|              CO(GT)|         PT08_S1(CO)|
NMHC(GT)|          C6H6(GT)|     PT08_S2(NMHC)|           NOx(GT)|
PT08_S3(NOx)|NO2(GT)|PT08_S4(NO2)|       PT08_S5(O3)|   T|  RH|    AH|
datetime|           CO(GT)_tn|      PT08_S1(CO)_tn|         NMHC(GT)_tn|
C6H6(GT)_tn|    PT08_S2(NMHC)_tn|          NOx(GT)_tn|     PT08_S3(NOx)_tn|
NO2(GT)_tn|     PT08_S4(NO2)_tn|      PT08_S5(O3)_tn|               T_tn|
RH_tn|               AH_tn|
+----------+--------+-------------------+-------------------+----------------
-+-----------------+-----------------+-----------------+-----------------+--
-----+-----------+-----------------+----+----+------+------------------+-----
--------------+-------------------+-------------------+-------------------+-
-----------------+-------------------+-------------------+-----------------
--+-------------------+-------------------+-------------------+-------------
------+-------------------+
|2004-03-11|07:00:00| 0.09531017980432493|8.741258741258741E-4|
3.367295829986474|1.1631508098056809| 6.502790045915623| 4.584967478670572|
7.306531398939505|     82|        1339|
6.593044534142437|10.2|59.6|0.7417|2004-03-11 07:00:00|[0.5017437352800399]| [0.
363776926590997]|[0.2768057208621973]|[0.5367600344560551]|[0.31618602313685…|
[0.5891364591936437]|[0.7225845253979329]|[0.23668639053254…|[0.35431654676258
…|[0.4907029127478999]|[0.2602150537634409]|[0.6339622641509435]|[0.2721986023
5547…|
|2004-03-14|14:00:00|  0.5877866649021191|8.285004142502071E-4|
4.430816798843313|2.0149030205422647| 6.778784897685177| 4.634728988229636|
7.00669522683704|    102|        1490|
6.77078942390898|21.4|30.2|0.7616|2004-03-14
14:00:00|[0.6047911844216312]|[0.32054639394153…|[0.48391959036707…|
[0.668676186986218]|[0.4734909697575655]| [0.596669263203117]|  [0.5811616616209
4]|[0.2958579881656805]|[0.4222122302158274]|[0.5636975242258475]|[0.50107526881
72043]|[0.2641509433962264]|[0.2819234716317256]|
|2004-03-15|09:00:00|  2.0918640616783932|5.099439061703213E-4|
6.42648845745769|3.6027767550605247| 7.438971592395862| 6.169610732491456|
6.285998094508865|    149|        2665|
7.688913336864796|14.8|54.3|0.9076|2004-03-15
09:00:00|[0.9195094374536361]|[0.01871124398857…|[0.8725638123245671]| [0.9146
00049725836]|[0.8497683970549277]|[0.8290167835997715]|[0.24123252080037…|[0.4
349112426035503]|[0.9505395683453238]|[0.9407440789979543]|[0.35913978494623…|
[0.5672955974842767]|[0.35327175878414…|
|2004-03-19|22:00:00|  0.7884573603642703| 8.51063829787234E-4|
5.384495062789089|2.2082744135228043| 6.851184927493743| 4.962844630259907|
6.806829360392176|    116|        1604|
6.985641817639208|14.7|57.6|0.9573|2004-03-19 22:00:00|[0.6467802009748529]|[0.3
4192543263429…|[0.6696422994555633]|[0.6986248150611968]|[0.5147558076752067]|
```

```
[0.6463387938455281]|[0.4868915040021876]|[0.33727810650887…|[0.47347122302158
…|[0.6519310989072002]|[0.35698924731182…|
[0.608805031446541]|[0.37755949762986…|
|2004-03-22|17:00:00|  0.9555114450274365|8.680555555555555E-4|
5.220355825078325| 2.517696472610991| 6.967909201801884| 4.927253685157205|
6.833031732786201|    103|        1606|
6.745236349484362|20.2|28.5|0.6682|2004-03-22 17:00:00|[0.6817351639307071]| [0.
358025245273989]|[0.6376772384409238]|[0.7465469302354862]|[0.5812835186874172]|
[0.6409511033122985]|[0.49925030152728…|[0.29881656804733…|[0.47437050359712
…|[0.5532036262457597]|[0.4752688172043011]|[0.2427672955974843]|
[0.236280115330108]|
|2004-03-24|22:00:00|  0.5306282510621704|9.551098376313276E-4|
4.574710978503383|1.7749523509116738| 6.695798917058491| 4.770684624465665|
6.96979066990159|    108|        1435|
6.641182169740591|10.1|66.9|0.8248|2004-03-24 22:00:00|[0.5928311641538279]|[0.4
4050995871703…|[0.5119420565161559]|[0.6315135371736271]|[0.4261926047009324]|
[0.6172499724867389]|[0.5637549955351385]|[0.3136094674556213]|[0.39748201438848
…|[0.5104716274072605]|[0.25806451612903…|[0.7257861635220126]|[0.3128084836
0455…|
|2004-04-08|04:00:00|
0.7667458827323177|0.001215066828675577|3.6375861597263857|0.5877866649021191|
6.342121418721152|3.7612001156935624|7.2196420401307355|     57|        1263|
6.54965074223381| 8.3|75.6|0.8302|2004-04-08 04:00:00|[0.6422372177974467]|[0.68
68215404395698]|[0.32944302442411…|[0.44764981749066…|[0.22461196056865…|[
0.46443610042857…|[0.6816016717671722]|[0.16272189349112…|[0.32014388489208.
..|[0.4728823555150846]|[0.21935483870967…|[0.8352201257861634]|[0.31544739285
53976]|
|2004-04-29|01:00:00|                0.0|9.596928982725527E-4|
4.430816798843313|1.6486586255873816| 6.648984550024776|3.9318256327243257|
6.827629234502852|     61|        1495|
6.875232087276577|16.8|59.5|1.1269|2004-04-29 01:00:00|[0.48180071017476…|[0.4
448524498004142]|[0.48391959036707…|[0.6119536439048157]|[0.39951047110276…|
[0.4902650712575725]|[0.49670212072099…|[0.17455621301775…|[0.42446043165467
63]|[0.6065890618911676]|[0.4021505376344086]|[0.6327044025157232]|[0.4604407955
822704]|
|2004-04-29|08:00:00|  1.9740810260220096| 5.64652738565782E-4|
7.029087564149662|3.5890591188317256| 7.432483807917119| 5.908082938168931|
6.133398042996649|    125|        2572|
7.591861714889934|16.1|56.5|1.0274|2004-04-29 08:00:00|[0.8948641158112572]|[0.0
7054835360962…|[0.9899161132211918]|[0.9124755142443881]|[0.8460706447795797]|
[0.789427196484792]|[0.1692560939868737]|[0.36390532544378…|[0.908723021582733
9]|[0.9008878251913641]|[0.38709677419354…|[0.5949685534591195]|[0.41181644920
099…|
|2004-05-03|03:00:00|  -0.916290731874155|0.001146788990825…|
5.384495062789089|0.5306282510621704| 6.335054251498059|5.5053315359323625|
7.157735484249907|    113|        1385|
6.202535517187923|15.6|65.3|1.1484|2004-05-03 03:00:00|[0.29007293138962…|[0.6
221276763898129]|[0.6696422994555633]|[0.4387973391014755]|[0.22058398556962…|
```

```
[0.7284594444115223]|[0.6524023848437608]|[0.32840236686390…|
[0.375]|[0.33033231639112…|[0.37634408602150…|[0.7056603773584905]|
[0.470947563895812]|
|2004-05-06|10:00:00|  0.9162907318741551|9.587727708533077E-4|
5.384495062789089|2.5257286443082556|  6.97166860472579| 5.081404364984463|
6.670766320845874|    114|        1692|
6.921658184151129|20.2|37.7|0.8833|2004-05-06 10:00:00| [0.673528488959907]|[0.4
439806208139726]|[0.6696422994555633]|[0.7477909225297339]|[0.5834262132792876]|
[0.664286144177318]|[0.42271504199658…|[0.3313609467455621]|[0.513039568345323
7]|[0.6256548978105381]|[0.4752688172043011]|[0.35849056603773…|[0.34139666715
535…|
|2004-05-09|17:00:00|  0.7667458827323177|0.001033057851239…|
5.384495062789089|1.6863989535702288| 6.665683717782408|5.5053315359323625|
6.918695219020472|    113|        1446|
6.309918278226516|17.8|44.4|0.8967|2004-05-09
17:00:00|[0.6422372177974467]|[0.5143664010726597]|[0.6696422994555633]| [0.6177
98722790397]|[0.4090282634669458]|[0.7284594444115223]|[0.5396549513713043]|[0.3
2840236686390…|[0.40242805755395…|[0.37443126591218…|[0.42365591397849…|
[0.44276729559748…|[0.3479450715926306]|
|2004-05-15|02:00:00|                 0.0|0.001069518716577…|
5.384495062789089|1.6863989535702288| 6.666956792429207| 4.007333185232471|
6.871091294610546|     71|        1389|
6.860663671448287|17.4|43.4|0.8553|2004-05-15
02:00:00|[0.48180071017476…|[0.5489133981596294]|[0.6696422994555633]| [0.6177
98722790397]|[0.4097538601151253]|[0.5016952629723911]|[0.5172017454426527]|[0.2
0414201183431…|[0.3767985611510792]|[0.6006062409243768]|[0.4150537634408602]|
[0.4301886792452831]|[0.32771343400283…|
|2004-06-05|12:00:00| 0.33647223662121284|0.001094091903719…|
5.384495062789089|2.0794415416798357| 6.802394763324311| 4.454347296253507|
6.917705609835305|     68|        1596|
6.559615237493242|30.7|26.0|1.1287|2004-06-05 12:00:00|[0.5522053015409852]|[0.5
721967134830137]|[0.6696422994555633]|[0.6786716684080888]|[0.4869475564666069]|
[0.5693634209046518]|[0.5391881852569793]|[0.1952662721893491]|[0.46987410071942
45]|[0.4769744813758698]|[0.7010752688172044]|[0.21132075471698…|[0.4613204319
9921…|
|2004-06-07|04:00:00|  0.7667458827323177|0.001377410468319…|
5.384495062789089|0.7419373447293773|6.3818160174060985| 3.091042453358316|
7.141245122350491|     34|        1386| 6.361302477572996|17.2|57.9|
1.127|2004-06-07 04:00:00|[0.6422372177974467]|[0.8406435957829289]|[0.669642299
4555633]|[0.47152409023853…|[0.24723613883585…|[0.36298889011381…|[0.64462
44233383778]|[0.09467455621301…|[0.3754496402877698]|[0.3955332490540239]|
[0.410752688172043]|[0.6125786163522013]|[0.46048966427210…|
|2004-06-21|13:00:00|  0.1823215567939546|0.001193317422434…|
5.384495062789089|1.9740810260220096|6.7661917146603505| 4.189654742026425|
6.912742820493176|     76|        1454|
6.29156913955832|30.1|18.5|0.7747|2004-06-21 13:00:00|[0.5199502907530351]|[0.66
62137828590619]|[0.6696422994555633]|[0.6623538312730904]|[0.4663134083979822]|[
0.5292947583695472]|[0.5368474007006521]|[0.21893491124260…|[0.40602517985611.
```

```
..|[0.3668958129953714]|[0.6881720430107527]|[0.11698113207547…|[0.28832526999
951…|
|2004-06-29|17:00:00|   1.308332819650179|7.593014426727411E-4|
5.384495062789089| 3.068052935133617| 7.198931240688173| 5.298317366548036|
6.385194398997726|    198|        2132|
7.216709486709457|37.1|26.9|1.6635|2004-06-29 17:00:00|[0.7555607043527716]|[0.2
549797474191655]|[0.6696422994555633]|[0.8317840521835599]|[0.7129559367133801]|
[0.69712202762144]|[0.28802015605645…|[0.5798816568047337]|[0.7108812949640289
]|[0.7468238114406026]|[0.8387096774193549]|[0.22264150943396…|[0.722670185212
3344]|
|2004-07-04|13:00:00|-0.35667494393873245|0.001131221719457…|
5.384495062789089|1.3609765531356006| 6.555356891810665|3.5263605246161616|
6.98100574072173|     46|        1394|6.1903154058531475|37.9|18.8|
1.222|2004-07-04 13:00:00|[0.4071688358461722]|[0.6073775470760396]|[0.669642299
4555633]|[0.5673985354221223]|[0.3461468170955919]|[0.42888652347791…| [0.5690
44775689829]|[0.1301775147928994]|[0.3790467625899281]|[0.3253138751843419]|[0.8
559139784946237]|[0.1207547169811321]|[0.5069149196110051]|
|2004-07-08|02:00:00| -0.5108256237659907|0.001039501039501…|
5.384495062789089|1.7578579175523736| 6.688354713946762|3.7612001156935624|
6.748759547491679|     60|        1507|  6.97166860472579|27.5|34.0|
1.231|2004-07-08 02:00:00|[0.37491382505822…|[0.5204713797678623]|[0.669642299
4555633]|[0.6288660161575185]|[0.42194973580096…|[0.46443610042857…|[0.45950
188259061…|[0.17159763313609…|[0.4298561151079137]|[0.6461927103152509]|[0.6
322580645161291]|[0.31194968553459…|[0.5113131016957435]|
|2004-07-12|05:00:00| -0.6931471805599453|0.001197604790419…|
5.384495062789089|1.1939224684724346|  6.510258340523 15|3.7612001156935624|
6.959398512133975|     43|
1423|6.4425401664681985|20.0|51.8|1.1967|2004-07-12 05:00:00|[0.33676424447995..
.|[0.6702761024970877]|[0.6696422994555633]|[0.5415258322633532]|[0.320442623131
5783]|[0.46443610042857…|[0.5588533564099695]|[0.12130177514792…|[0.39208633
09352518]|[0.42889518450015…|[0.47096774193548…|[0.5358490566037736]|[0.4945
511410839075]|
+----------+--------+------------------+------------------+-----------------
-+------------------+-----------------+-----------------+------------------+--
-----+------------+------------------+----+----+------+------------------+-----
--------------+------------------+------------------+------------------+-
------------------+------------------+------------------+-----------------
--+------------------+------------------+------------------+-------------
------+--------------------+
only showing top 20 rows
```

[ ]:
```
import airpolution_model_v1
from pyspark.ml.regression import DecisionTreeRegressor, RandomForestRegressor

model_list = [
    DecisionTreeRegressor(labelCol="T", featuresCol="features"),
```

```
    RandomForestRegressor(labelCol="T", featuresCol="features")
]

df_model = airpolution_model_v1.model_fn(model_list, df_processing_normalize,␣
 ↪assembler_transform=True)
```

```
Vectorizing the data
Splitting the data

, DecisionTreeRegressor_f408023cb757
RMSE:  2.1341901411776214
MAE:  1.7201617320661717
R-squared:  0.9374457065284536



, RandomForestRegressor_c187b1ee5ace
RMSE:  2.7504634164500135
MAE:  2.1283648685140153
R-squared:  0.8961030855030346
```

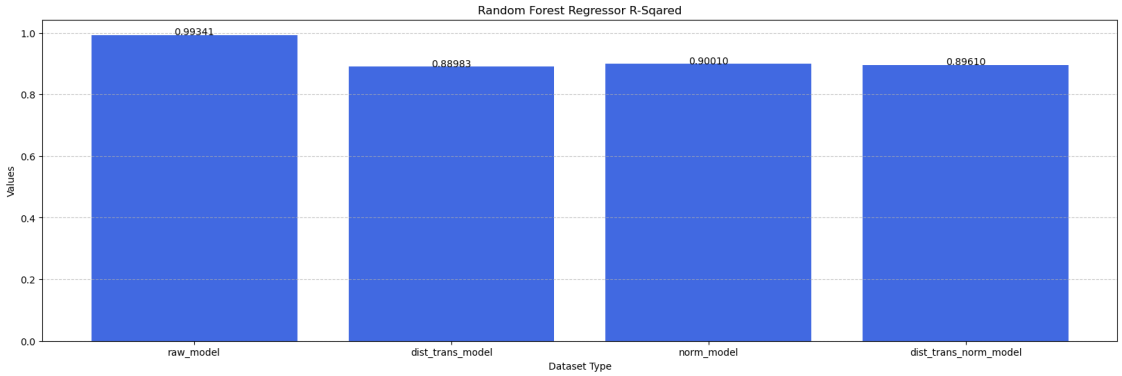
[ ]:
```
result_r2_json_decision = {
    "raw_model": 0.9956220651287081,
    "dist_trans_model": 0.9415494509096733,
    "norm_model": 0.9455756805363574,
    "dist_trans_norm_model": 0.9374457065284536
}

result_r2_json_random = {
    "raw_model": 0.9934124223898312,
    "dist_trans_model": 0.8898315601129945,
    "norm_model": 0.9000956171506417,
    "dist_trans_norm_model": 0.8961030855030346
}
```

[ ]:
```
import matplotlib.pyplot as plt


fig, ax = plt.subplots(figsize=(20, 6))

ax.bar(result_r2_json_decision.keys(), result_r2_json_decision.values(),␣
 ↪color='royalblue')

for i, v in enumerate(result_r2_json_decision.values()):
    ax.text(i, v, f"{v:.5f}", ha='center', fontsize=10)
```

```
ax.set_title('Decision Tree Regressor R-Squared')
ax.set_xlabel('Dataset Type')
ax.set_ylabel('Values')
ax.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()
```

From the graph above we can see that the raw data has the higher value of R-squared and the second highest value in norm_model Decision Tree Regressor model

[ ]:
```
fig, ax = plt.subplots(figsize=(20, 6))

ax.bar(result_r2_json_random.keys(), result_r2_json_random.values(),␣
 ↪color='royalblue')

for i, v in enumerate(result_r2_json_random.values()):
    ax.text(i, v, f"{v:.5f}", ha='center', fontsize=10)

ax.set_title('Random Forest Regressor R-Sqared')
ax.set_xlabel('Dataset Type')
ax.set_ylabel('Values')
ax.grid(axis='y', linestyle='--', alpha=0.7)

plt.show()
```



From the graph above we can see that the raw model has the highest value of R-squared and the norm_model has the second highest value from Random Forest Regressor model

5 Conclusion

From this projects we know that raw data has the best performance based on the R-squared evalutaion matrix on both model, then the only normalize data sit on the second position.