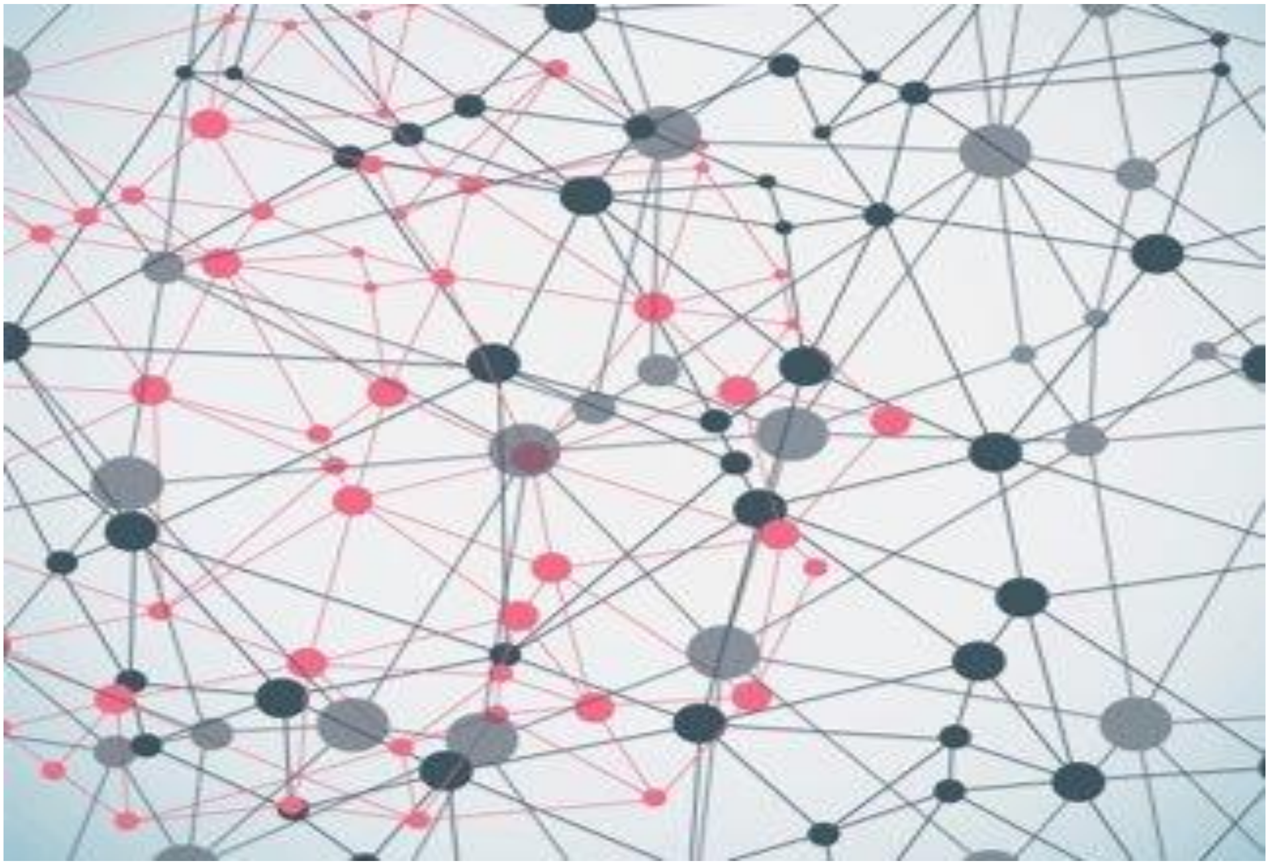
AIR QUALITY ANALYSIS AND PREDICTION IN TAMIL NADU



PROJECT REPORT PHASE- 2
SUBMITTED BY,
MICHEAL RAJ.F
REG NO:9617211060308

INTRODUCTION

Air pollution is one of the greatest environmental risk to health. By reducing air pollution levels, countriescan reduce the burden of disease from stroke, heart disease, lung cancer, and both chronic and acute respiratory diseases, including asthma.

Here we are studied about the air quality analysis methods in Tamil Nadu

Content for Project Phase 2 :

For analyzing data, we need some libraries. In this section, we are importing all the required libraries like pandas NumPy, matplotlib, plotly, seaborn, and word cloud that are required for data analysis. Check the below code to import all the required libraries

Data Source:

A good data source for credit card fraud detection should be accurate,complete, Covering the geographic area of interest, Accessible.

Dataset Link

https://tn.data.gov.in/resource/location-wise-daily-ambient-air-quality-tamil-nadu-year-2014

EXPLORATORY DATA ANALYSIS

Exploratory data analysis is performed on the raw data. The insights gained from the analysis helps to identify the pre- processing tasks that need to be performed to form the dataset for building the air quality prediction model.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

from sklearn.preprocessing import
LabelEncoder
from sklearn.model_selection import
train_test_split
from sklearn.linear_model import
LinearRegression
from sklearn.tree import
DecisionTreeRegressor
```

```
from sklearn.ensemble import
RandomForestRegressor
from sklearn import metrics
from sklearn.metrics import
mean_absolute_error,mean_squared_error,r2_s
core
from sklearn.metrics import
accuracy_score,confusion_matrix
```

```
df=pd.read_csv('../input/india-air-
quality-
data/data.csv',encoding='unicode_escap
e')
# Reading the dataset
```

Data Understanding

```
df.head()
# Loading the dataset
```

```
output
```

	stn_co	sampling_date	state	location	agency	type	so2	no2	rspm	spm	location_monitoring_station	pm2_5	date

	de												
0	150.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	4.8	17.4	NaN	NaN	NaN	NaN	1990-02-01
1	151.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	3.1	7.0	NaN	NaN	NaN	NaN	1990-02-01
2	152.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.2	28.5	NaN	NaN	NaN	NaN	1990-02-01
3	150.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.3	14.7	NaN	NaN	NaN	NaN	1990-03-01
4	151.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	4.7	7.5	NaN	NaN	NaN	NaN	1990-03-01

```
df.shape
# As we can see that there are 4,35,742 rows and 13 columns in the dataset
```

```
(435742, 13)
```

```
df.info()
# Checking the over all information on the dataset.
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 435742 entries, 0 to 435741
Data columns (total 13 columns):
 #   Column                       Non-Null Count   Dtype  
---  ------                       --------------   -----  
 0   stn_code                     291665 non-null  object 
 1   sampling_date                435739 non-null  object 
 2   state                        435742 non-null  object 
 3   location                     435739 non-null  object 
 4   agency                       286261 non-null  object 
 5   type                         430349 non-null  object 
 6   so2                          401096 non-null  float64
 7   no2                          419509 non-null  float64
 8   rspm                         395520 non-null  float64
 9   spm                          198355 non-null  float64
 10  location_monitoring_station  408251 non-null  object 
 11  pm2_5                        9314 non-null    float64
 12  date                         435735 non-null  object 
dtypes: float64(5), object(8)
memory usage: 43.2+ MB
```

In [7]:

```
df.isnull().sum()
# There are a lot of missing values present in the dataset
```

Out[7]:

```
stn_code                        144077
sampling_date                        3
state                                0
location                             3
agency                          149481
type                              5393
so2                              34646
no2                              16233
rspm                             40222
spm                             237387
location_monitoring_station      27491
pm2_5                           426428
date                                 7
dtype: int64
```

In [8]:

```
df.describe()
# Checking the descriptive stats of the numeric values present in the data like mean, standard deviation, min values and max value present in the data
```

Out[8]:

	so2	no2	rspm	spm	pm2_5
count	401096.000000	419509.000000	395520.000000	198355.000000	9314.000000
mean	10.829414	25.809623	108.832784	220.783480	40.791467
std	11.177187	18.503086	74.872430	151.395457	30.832525
min	0.000000	0.000000	0.000000	0.000000	3.000000
25%	5.000000	14.000000	56.000000	111.000000	24.000000
50%	8.000000	22.000000	90.000000	187.000000	32.000000
75%	13.700000	32.200000	142.000000	296.000000	46.000000

max	909.000000	876.000000	6307.033333	3380.000000	504.000000

In [9]:

```
df.nunique()
# These are all the unique values present in the dataframe
```

Out[9]:

```
stn_code                          803
sampling_date                    5485
state                              37
location                          304
agency                             64
type                               10
so2                              4197
no2                              6864
rspm                             6065
spm                              6668
location_monitoring_station       991
pm2_5                             433
date                             5067
dtype: int64
```

In [10]:

```
df.columns
# These are all the columns present in the dataset.
```

Out[10]:

```
Index(['stn_code', 'sampling_date', 'state', 'location', 'agency', 'type',
       'so2', 'no2', 'rspm', 'spm', 'location_monitoring_station',
'pm2_5',
       'date'],
      dtype='object')
```

stn_code (station code) sampling_date (date of sample collection) state (Indian State) location (location of sample collection) agency type (type of area) so2 (sulphur dioxide concentration) no2 (nitrogen dioxide concentration) rspm (respirable suspended particualte matter concentration) spm (suspended particulate matter) location_monitoring_station pm2_5 (particulate matter 2.5) date (date)

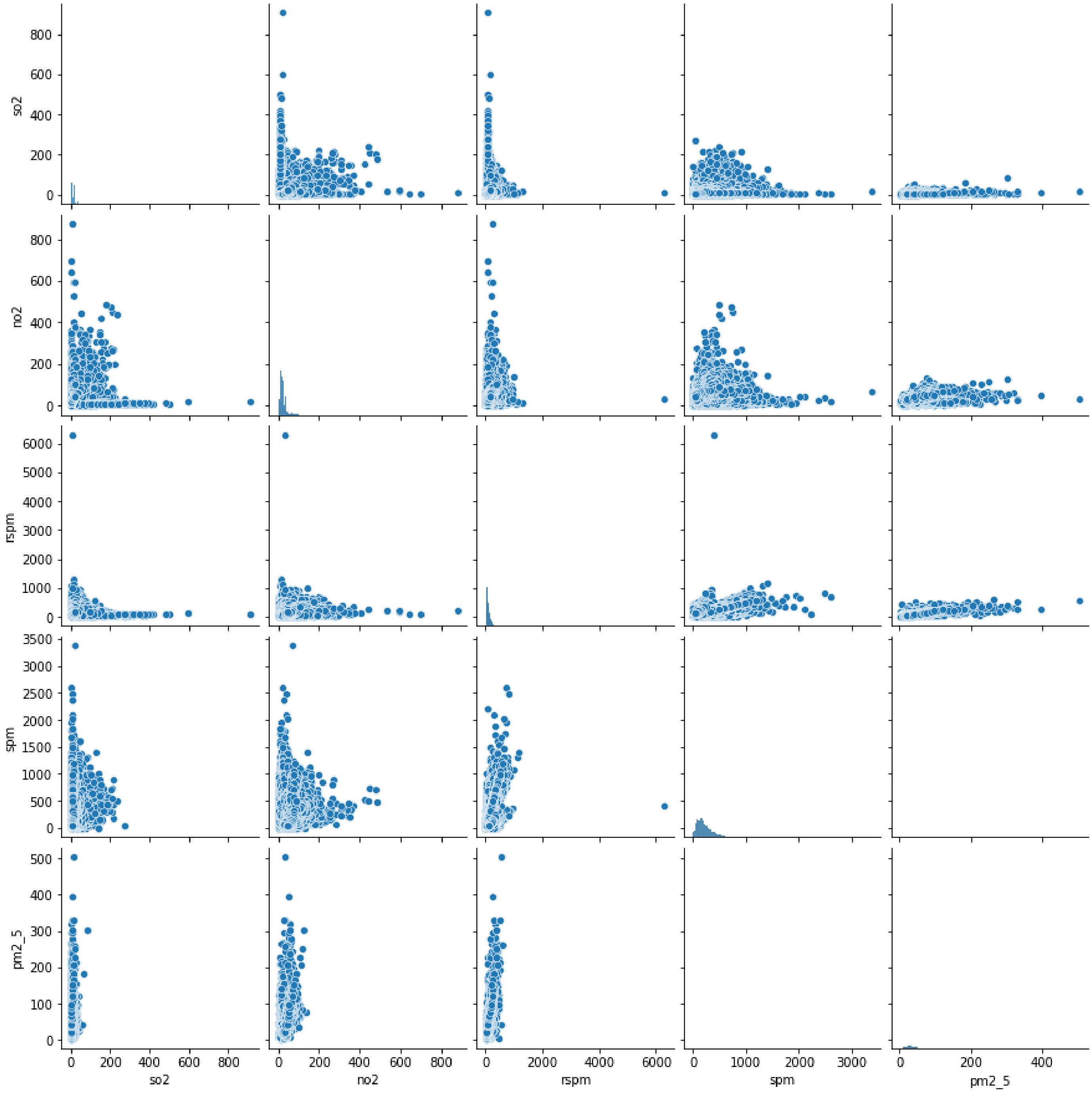
Data Visualization

In [11]:

```
sns.pairplot(data=df)
```

Out[11]:

```
<seaborn.axisgrid.PairGrid at 0x7fd7799bb690>
```


so2
800
600
400
200
0
no2
800
600
400
200
0
rspm
6000
5000
4000
3000
2000
1000
0
spm
3500
3000
2500
2000
1500
1000
500
0
pm2_5
500
400
300
200
100
0
0
200
400
600
800
so2
0
200
400
600
800
no2
0
2000
4000
6000
rspm
0
1000
2000
3000
spm
0
200
400
pm2_5

```
df['type'].value_counts()
# Viewing the count of values present
in the type column
```

Out[14]:

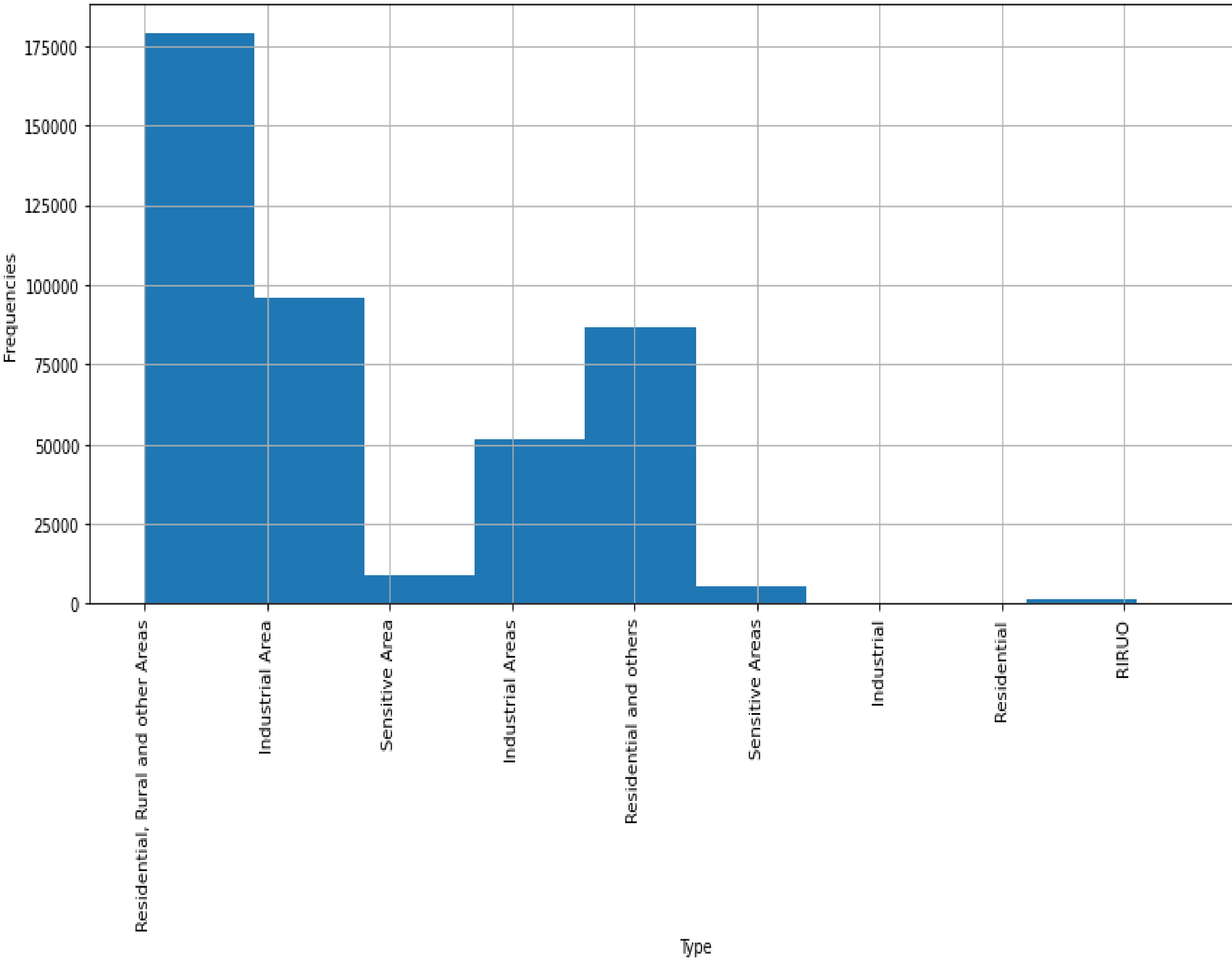
```
Residential, Rural and other Areas    179014
Industrial Area                        96091
Residential and others                 86791
Industrial Areas                       51747
Sensitive Area                          8980
Sensitive Areas                         5536
RIRUO                                   1304
Sensitive                                495
Industrial                               233
Residential                              158
Name: type, dtype: int64
```

In [15]:

```
plt.figure(figsize=(15, 6))
plt.xticks(rotation=90)
df.type.hist()
plt.xlabel('Type')
plt.ylabel('Frequencies')
plt.plot()
# The visualization shows us the count
of Types present in the dataset.
```

Out[15]:

```
[]
```


Frequencies
175000
150000
125000
100000
75000
50000
25000
0
Residential, Rural and other Areas
Industrial Area
Sensitive Area
Industrial Areas
Residential and others
Sensitive Areas
Industrial
Residential
RIRUO
Type

CONCLUTION

In conclusion, ambient air pollution is a health hazard. It is a global challenge, as evidence shows that adverse effects still exist even at relatively low air pollutant concentrations, and so no threshold values for classical air pollutants can be established based on the available data.