Air Quality Analysis And Prediction In Tamil Nadu

Rapid industrialization, vehicular emissions, and natural factors contribute to deteriorating air quality, affecting the well-being of residents. We aim to develop a comprehensive air quality analysis and prediction system for Tamil Nadu, leveraging machine learning and environmental data.

INOVATIVE SOLUTION

Developing innovative solutions for air pollution analysis and prediction in Tamil Nadu, or any region, is crucial for addressing the growing environmental and health challenges associated with poor air quality.

Here are some innovative ideas and technologies that can be implemented:

- Air Quality Monitoring Network Expansion: Expanding the existing air quality monitoring network in Tamil Nadu by deploying more sensors across urban and rural areas. These sensors can provide real-time data, which can be used for accurate pollution analysis.
- Satellite Data Integration: Incorporate satellite data and remote sensing technology to monitor air quality on a broader scale. Satellite imagery can provide valuable insights into regional and seasonal trends in air pollution.

- Community Engagement: Encourage community involvement in air quality monitoring and solutions. Establish local committees or organizations dedicated to addressing air pollution issues.

- Early Warning Systems: Create an early warning system that can alert residents and authorities to upcoming periods of poor air quality, allowing people to take preventive measures.

- Collaborative Research: Foster collaboration between government agencies, research institutions, and private companies to develop and implement innovative solutions for air quality analysis and prediction.

Data Visualization Tools: Develop user-friendly data visualization tools that make air quality information easily accessible to the public, enabling individuals and communities to make informed decisions.

Implementing a combination of these strategies and technologies can significantly improve air pollution analysis and prediction in Tamil Nadu, leading to better-informed decisions and a healthier environment for its residents.

DATA SET: https://www.kaggle.com/datasets/seshupavan/air-pollution-data-of-india-2020-2023/

TEAM DETAILS

- Project Manager: Kanishka.V
- Data Analyst: Dhrasika.R
- Data Engineer: Kanishka.M
- Data Scientist: Krishnaveni.N
- Visualization specialist: Harini.M

Steps in Data Preprocessing Step 1: Import the necessary libraries

```
# importing libraries
import pandas as pd
import scipy
import numpy as np
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
import matplotlib.pyplot as plt
```

Step 2: Load the dataset

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
# Load the dataset
df = pd.read_csv('/content/gdrive/MyDrive/ADS_PHASE 3.csv')
print(df.head())
```

```
   Benzene  Eth-Benzene  MP-Xylene      BP  O Xylene    PM10  PM2.5     RH  \
0     1.08         0.04       0.00  754.05      2.50  140.23  90.62  42.51   
1     0.83         0.03       0.00  754.28      1.74  124.91  61.11  28.34   
2     1.38         0.27       0.06  754.49      2.39  114.27  70.89  36.48   
3     1.97         0.47       0.12  754.28      3.51  128.15  78.52  43.68   
4     1.80         0.75       0.20  754.00      4.00  122.36  70.48  51.57   

       SR   Temp      WD    WS    CO    NH3     NO    NO2    NOx  Ozone    SO2  
0  125.03  17.90  119.19  0.97  0.51  20.53   4.29  22.95  27.24  44.36   4.97  
1  148.95  20.04   71.50  1.21  0.53  17.37   2.80  25.59  28.38  53.04   5.59  
2  131.87  18.31  147.10  1.00  0.78  18.45   6.85  30.91  37.75  41.94  10.22  
3  129.32  18.56  182.79  1.06  0.81  22.52   7.36  29.05  36.41  44.15  30.99  
4  145.73  18.81  183.47  0.91  0.96  19.14  13.15  28.60  41.74  37.13  15.78  
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 298 entries, 0 to 297
Data columns (total 19 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Benzene      298 non-null    float64
 1   Eth-Benzene  298 non-null    float64
 2   MP-Xylene    298 non-null    float64
 3   BP           298 non-null    float64
 4   O Xylene     298 non-null    float64
 5   PM10         298 non-null    float64
 6   PM2.5        298 non-null    float64
 7   RH           298 non-null    float64
 8   SR           298 non-null    float64
 9   Temp         298 non-null    float64
 10  WD           298 non-null    float64
 11  WS           298 non-null    float64
 12  CO           298 non-null    float64
 13  NH3          298 non-null    float64
 14  NO           298 non-null    float64
 15  NO2          298 non-null    float64
 16  NOx          298 non-null    float64
 17  Ozone        298 non-null    float64
 18  SO2          298 non-null    float64
dtypes: float64(19)
memory usage: 44.4 KB
```

```
df.head()
```

	Benzene	Eth-Benzene	MP-Xylene	BP	O Xylene	PM10	PM2.5	RH	SR	Temp	WD	WS	CO	NH3	NO	NO2	NOx
0	1.08	0.04	0.00	754.05	2.50	140.23	90.62	42.51	125.03	17.90	119.19	0.97	0.51	20.53	4.29	22.95	27.24
1	0.83	0.03	0.00	754.28	1.74	124.91	61.11	28.34	148.95	20.04	71.50	1.21	0.53	17.37	2.80	25.59	28.38
2	1.38	0.27	0.06	754.49	2.39	114.27	70.89	36.48	131.87	18.31	147.10	1.00	0.78	18.45	6.85	30.91	37.75
3	1.97	0.47	0.12	754.28	3.51	128.15	78.52	43.68	129.32	18.56	182.79	1.06	0.81	22.52	7.36	29.05	36.41
4	1.80	0.75	0.20	754.00	4.00	122.36	70.48	51.57	145.73	18.81	183.47	0.91	0.96	19.14	13.15	28.60	41.74

```
df.tail()
```

	Benzene	Eth-Benzene	MP-Xylene	BP	O Xylene	PM10	PM2.5	RH	SR	Temp	WD	WS	CO	NH3	NO	NO2	NOx
293	1.31	0.22	0.08	757.14	2.57	178.71	81.02	29.44	207.84	28.43	210.70	1.01	1.01	23.42	6.62	32.33	38.13
294	0.64	0.01	0.00	756.64	2.24	158.81	76.39	32.40	214.16	28.21	132.54	1.20	0.62	21.50	8.26	25.42	33.68
295	0.71	0.07	0.03	756.01	2.03	138.19	63.19	30.18	202.04	28.75	172.23	1.18	0.72	19.76	8.66	27.63	36.29
296	0.89	0.10	0.04	755.72	2.51	144.54	58.81	29.82	206.48	29.82	214.90	1.36	0.83	19.35	9.01	27.79	36.80
297	1.10	0.20	0.08	756.84	2.82	152.39	62.50	34.85	167.95	28.89	169.89	0.97	0.91	17.60	8.92	33.32	42.24

STEP 3: check the null values

```
df.isnull().sum()
```

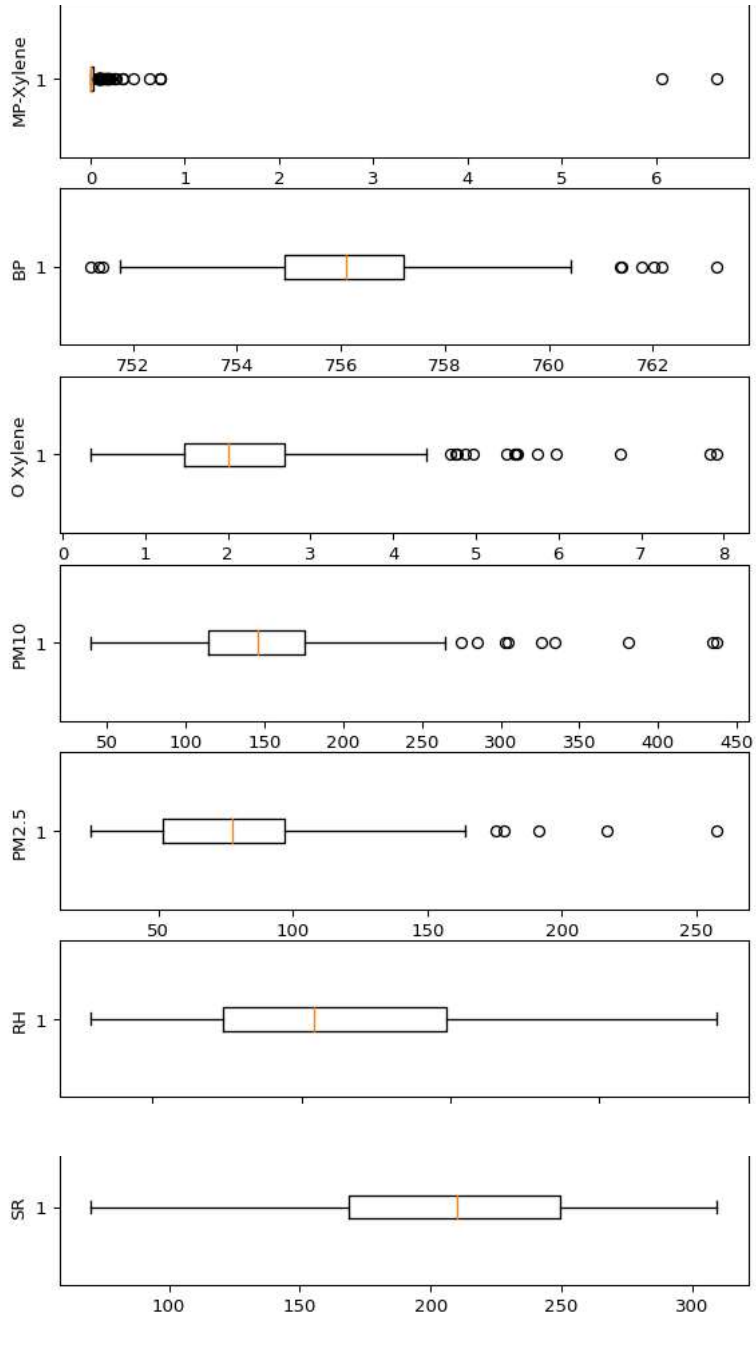
```
Benzene        0
Eth-Benzene    0
MP-Xylene      0
BP             0
O Xylene       0
PM10           0
PM2.5          0
RH             0
SR             0
Temp           0
WD             0
WS             0
CO             0
NH3            0
NO             0
NO2            0
NOx            0
Ozone          0
SO2            0
dtype: int64
```

```
df.describe()
```

SR	Temp	
298.000000	298.00000	2
208.083020	30.42802	1
50.891946	5.45345	
69.960000	16.67000	
168.785000	28.14250	1
210.255000	30.86500	2
249.402500	34.15250	2
309.610000	40.07000	2

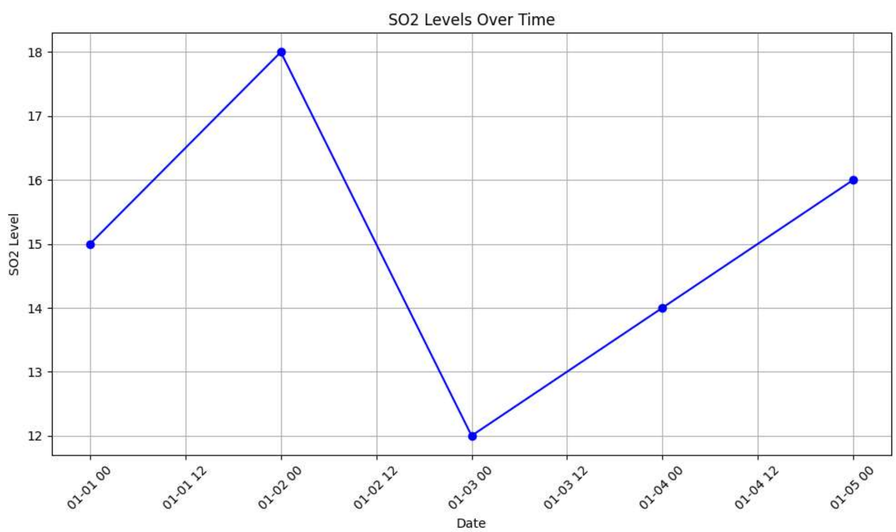
Step 5: Check the outliers

```
# Box Plots
fig, axs = plt.subplots(9,1,dpi=95, figsize=(7,17))
i = 0
for col in df.columns:
    axs[i].boxplot(df[col], vert=False)
    axs[i].set_ylabel(col)
    i+=1
plt.show()
```


MP-Xylene
BP
O Xylene
PM10
PM2.5
RH
SR

STEP 1: create visualization for SO2

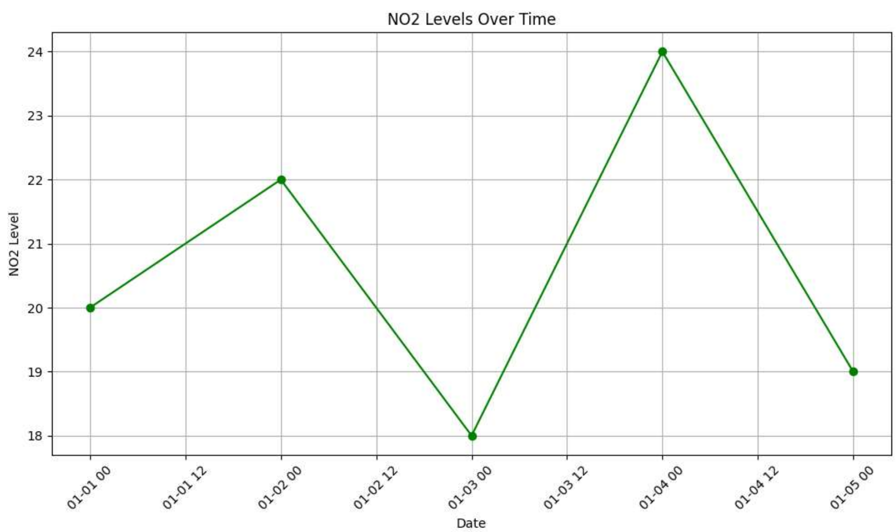
```
import matplotlib.pyplot as plt
import pandas as pd

# Sample data (replace this with your actual SO2 dataset)
data = {
    'Date': ['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04', '2023-01-05'],
    'SO2_Level': [15, 18, 12, 14, 16]  # Replace with your SO2 data
}

# Create a DataFrame from the data
df = pd.DataFrame(data)
df['Date'] = pd.to_datetime(df['Date'])  # Convert Date column to datetime format

# Plot the SO2 levels
plt.figure(figsize=(10, 6))
plt.plot(df['Date'], df['SO2_Level'], marker='o', linestyle='-', color='b')
plt.title('SO2 Levels Over Time')
plt.xlabel('Date')
plt.ylabel('SO2 Level')
plt.grid(True)

# Format the x-axis to display dates nicely
plt.xticks(rotation=45)

# Show the plot
plt.tight_layout()
plt.show()
```



STEP 2: create visualization for NO2

```
import matplotlib.pyplot as plt
import pandas as pd

# Sample data (replace this with your actual NO2 dataset)
data = {
    'Date': ['2023-01-01', '2023-01-02', '2023-01-03', '2023-01-04', '2023-01-05'],
    'NO2_Level': [20, 22, 18, 24, 19]  # Replace with your NO2 data
}
```

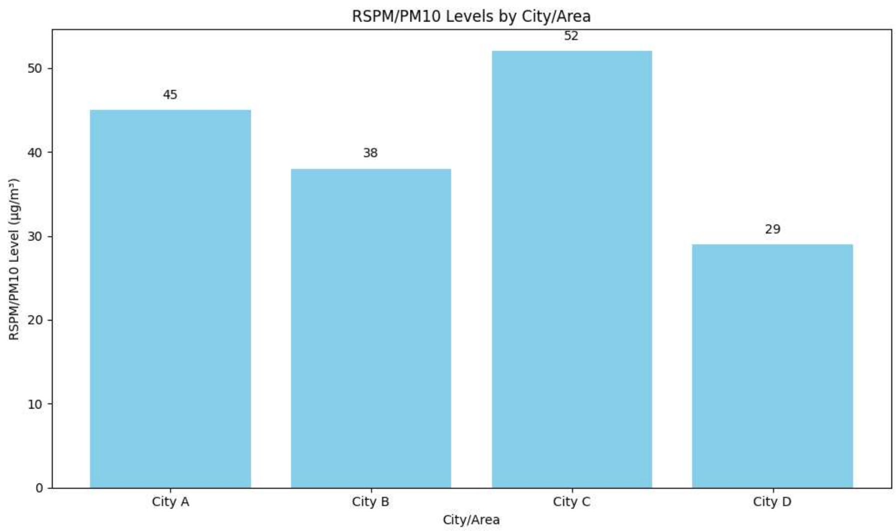
```
# Create a DataFrame from the data
df = pd.DataFrame(data)
df['Date'] = pd.to_datetime(df['Date'])  # Convert Date column to datetime format

# Plot the NO2 levels
plt.figure(figsize=(10, 6))
plt.plot(df['Date'], df['NO2_Level'], marker='o', linestyle='-', color='g')
plt.title('NO2 Levels Over Time')
plt.xlabel('Date')
plt.ylabel('NO2 Level')
plt.grid(True)

# Format the x-axis to display dates nicely
plt.xticks(rotation=45)

# Show the plot
plt.tight_layout()
plt.show()
```



STEP 3: create visualization for RSPM/PM10

```
import matplotlib.pyplot as plt

# Sample data (replace this with your actual RSPM/PM10 dataset)
categories = ['City A', 'City B', 'City C', 'City D']
rspm_pm10_levels = [45, 38, 52, 29]  # Replace with your RSPM/PM10 data

# Create a bar chart to visualize RSPM/PM10 levels
plt.figure(figsize=(10, 6))
plt.bar(categories, rspm_pm10_levels, color='skyblue')
plt.title('RSPM/PM10 Levels by City/Area')
plt.xlabel('City/Area')
plt.ylabel('RSPM/PM10 Level (µg/m³)')

# Add data labels above each bar
for i, level in enumerate(rspm_pm10_levels):
    plt.text(i, level + 1, str(level), ha='center', va='bottom')

# Show the plot
plt.tight_layout()
```

```
plt.show()
```




```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

STEP 4: grouping data

```
import pandas as pd

# Load your data into a Pandas DataFrame (replace 'data.csv' with your data file)
df = pd.read_csv('/content/drive/MyDrive/Excel 4.csv')

# Group the data by the desired column (e.g., 'City' or 'Monitoring Station')
grouped_data = df.groupby('CITIES')
```

STEP 5: calculating average

```
# Calculate the average SO2, NO2, and RSPM/PM10 levels for each group
averages = grouped_data[['SO2', 'NO2', 'RSPM/PM10']].mean()

# Display the calculated averages
print(averages)
```

```
                    SO2        NO2   RSPM/PM10
CITIES
Chennai         9.433333  23.036667  145.640000
Dindigul       22.880000  23.753333  129.780000
Erode          18.900000  27.065000  165.580000
Kanniyakumari  27.790000  34.330000  148.680000
Kodaikanal     12.130000  38.050000  171.910000
Madurai        13.897500  26.727500  156.572500
Salem          11.100000  30.060000  160.976667
Thanjavur       3.650000  30.790000  145.180000
Vellore        16.726667  23.116667  141.240000
madurai        10.220000  30.910000  114.270000
salem          30.990000  29.050000  128.150000
```

```
# Calculate the average SO2, NO2, and RSPM/PM10 levels for each group
averages = grouped_data[[ 'RSPM/PM10']].mean()

# Display the calculated averages
print(averages)
```

```
                RSPM/PM10
CITIES
Chennai        145.640000
Dindigul       129.780000
Erode          165.580000
Kanniyakumari  148.680000
Kodaikanal     171.910000
Madurai        156.572500
Salem          160.976667
Thanjavur      145.180000
Vellore        141.240000
madurai        114.270000
salem          128.150000
```