**AIR QUALITY ANALYSIS IN TAMIL NADU USING COGNOS**

**Project Title:** Air Quality Analysis In Tamil Nadu

**Phase 5 :** Project Documentation & Submission

**Topic :** Analysis by loading and preprocessing the air quality dataset using data manipulation libraries (e.g., pandas) and creating visualizations using data visualization libraries (e.g., Matplotlib, Seaborn) and visualization using cognos.

**TEAM MEMBERS**

Abinaya M -202341001

Alef Raja M -910021104004

Ashwini S -910021104005

Jayanthi E -911021104016

Kiruthiga K -911021104019



**Objectives:**

The primary objectives of this project are to analyze air quality data for Tamil Nadu in 2014 and provide insights into air pollution trends and pollution levels.

**Analysis Approach:**

This analysis involves several steps, including data download, data preprocessing, visualization, and trend identification.

**Step 1:** **Data Collection**

* Visit the provided dataset link: [Tamil Nadu Air Quality Dataset.](https://tn.data.gov.in/resource/location-wise-daily-ambient-air-quality-tamil-nadu-year-2014)
* Download the dataset in a format such as CSV or Excel.
* Ensure that the dataset is in a structured format, such as a CSV file, to facilitate analysis.

**Step 2: Import Necessary Libraries**

Before you start, make sure you have Python and the Pandas library installed. You can install Pandas using pip if it's not already installed.

**Input:** pip install pandas

**Step 3:** **Load the Dataset**

Assuming you have downloaded the dataset as a CSV file, you can load it into a Pandas DataFrame like this:

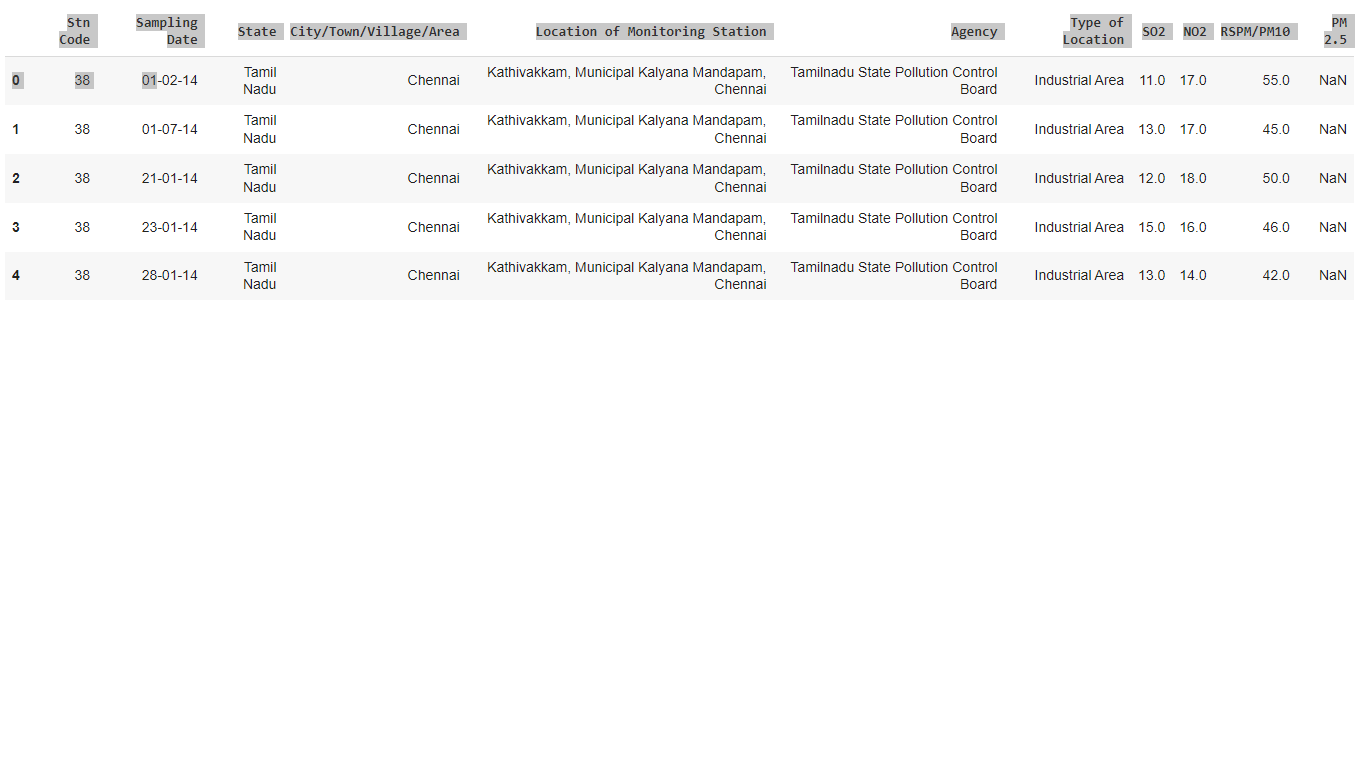
**Input:** import pandas as pd

from google.colab import files

upload = files.upload()

df = pd.read\_csv('/content/cpcb\_dly\_aq\_tamil\_nadu-2014.csv')

df.head()

**Output :**

**Step 4: Data Preprocessing**

Data preprocessing may involve various tasks, such as handling missing values, data cleaning, and feature engineering.

* Load the dataset into a Pandas DataFrame for further analysis.
* Address missing values, outliers, and inconsistencies in the data.
* Perform data cleaning, which may include removing duplicates, correcting data types, and ensuring data integrity.

Here are some common preprocessing steps:

**Handle Missing Values:** If there are missing values in the dataset, you can use Pandas to fill them or drop rows/columns with missing values.

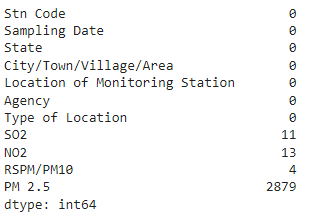
**Data Cleaning:** Check for and clean any outliers or incorrect data.

**Feature Engineering:** Create new features or transform existing ones if needed.

**Data Exploration**: You can use various Pandas functions to explore and understand the data. For example, you can use df.describe(), df.info(), and df['column\_name'].value\_counts() to gain insights into the dataset.

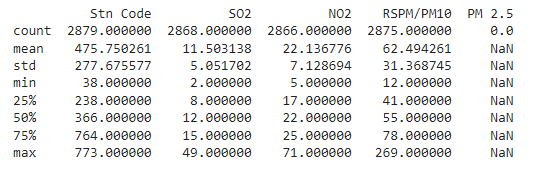
**Input: print(df.isnull().sum())**

**Output:**

****

**Input: print(df.describe())**

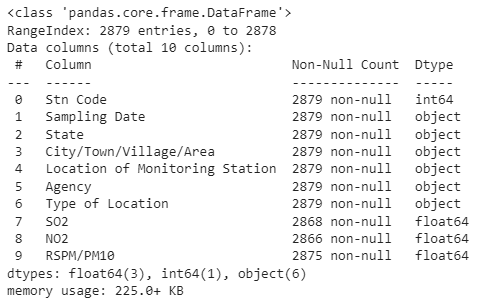
**Output:**

****

**Input: df.drop(["PM 2.5"],axis=1,inplace=True)**

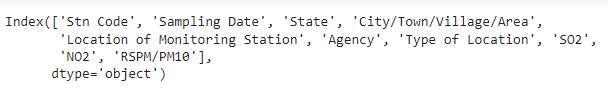
**Input:df.info()**

**Output:**

****

**Input:df.columns**

**Output:**

****

**Input:df.shape**

**Output:**



**Input: df.dtypes**

**Output:**



**Input: df.index**

**Output:**

****

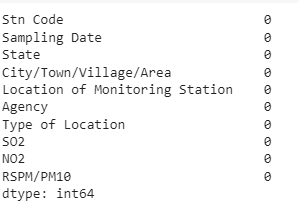
**Input:df["SO2"].fillna(0,inplace=True)**

**Input: df["NO2"].fillna(0,inplace=True)**

**Input: df["RSPM/PM10"].fillna(0,inplace=True)**

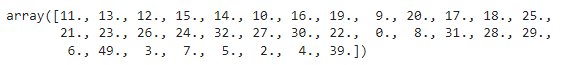
**Input: print(df.isnull().sum())**

**Output:**

****

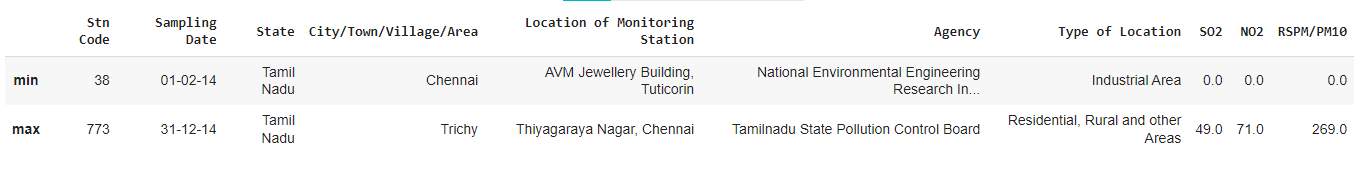
**Input: df['SO2'].unique()**

**Output:**

****

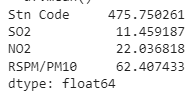
**Input: df.agg(['min', 'max'])**

**Output:**

****

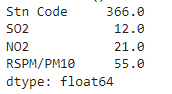
**Input: df.mean()**

**Output:**

****

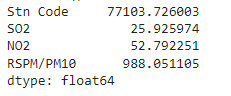
**Input: df.median()**

**Output:**

****

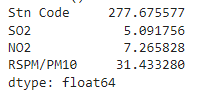
**Input:df.var()**

**Output:**

****

**Input: df.std()**

**Output:**

****

**Input: df.kurtosis()**

**Output:**

****

**Step 5: Save the Preprocessed Data**

If we want to save the preprocessed dataset for future use, we can save it to a new CSV file using to\_csv():

**Input: df.to\_csv('preprocessed\_dataset.csv', index=False)**

Here is the [preprocessed dataset](https://drive.google.com/file/d/1-Sc9WJgrSn4S8Pv2sJPPYL6Hlnmafxg1/view?usp=sharing)

**Step 6: Create Visualizations**

Use data visualization libraries such as Matplotlib and Seaborn to create visualizations that help convey your findings. Here are some visualization:

* Line Plot
* Scatter Plot
* Bar Plot
* Histogram
* Pie Chart

**Input:**

**%matplotlib inline**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**sns.set()**

**Input:**

**import seaborn as sns**

**corr = df.corr()**

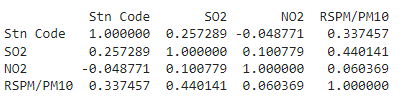
**print(corr)**

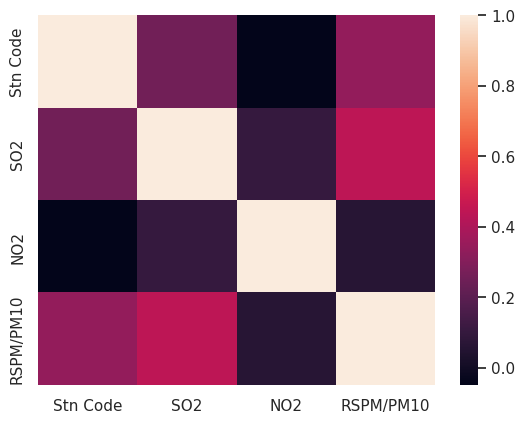
**sns.heatmap(corr,**

**xticklabels=corr.columns,**

**yticklabels=corr.columns)**

**Output:**

****

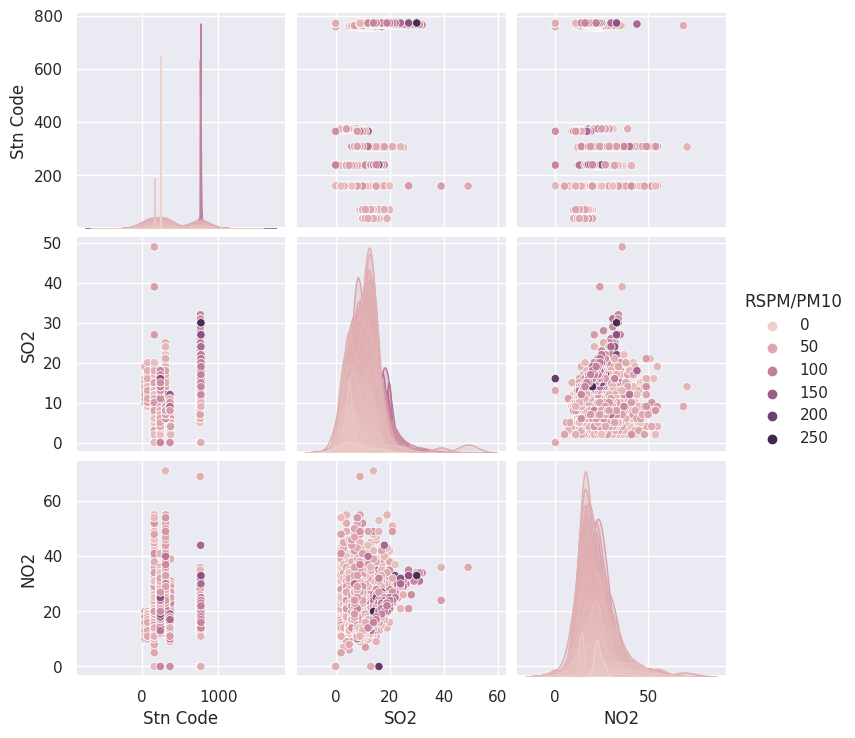
****

**Input:**

**import seaborn as sns**

**pd=sns.pairplot(df,hue = 'RSPM/PM10')**

**Output:**

****

**Input:**

**x=df['City/Town/Village/Area']**

**y=df['Location of Monitoring Station']**

**plt.plot(x,y)**

**plt.title("LINE PLOT",color="Green",fontsize=20)**

**plt.xlabel("City/Town/Village/Area",color="purple")**

**plt.ylabel("Location of Monitoring Station",color="purple")**

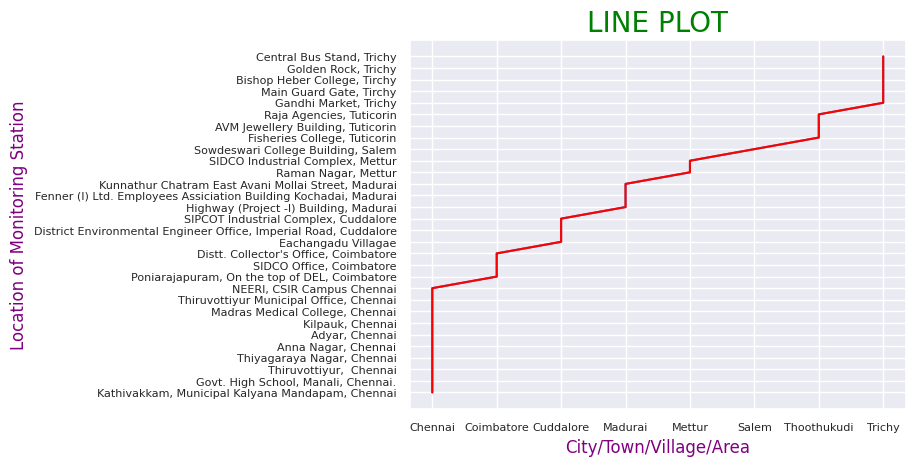
**plt.tick\_params(axis='x', which='major', labelsize=8)**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

**plt.plot(x,y,color="red")**

**plt.show()**

**Output:**

****

**Input:**

**x=df['SO2']**

**y=df['NO2']**

**z=df['RSPM/PM10']**

**plt.scatter(z,x,c='red')**

**plt.scatter(z,y,c='violet')**

**plt.title("SCATTER PLOT",color="Green",fontsize=20)**

**plt.xlabel("SO2",color="purple")**

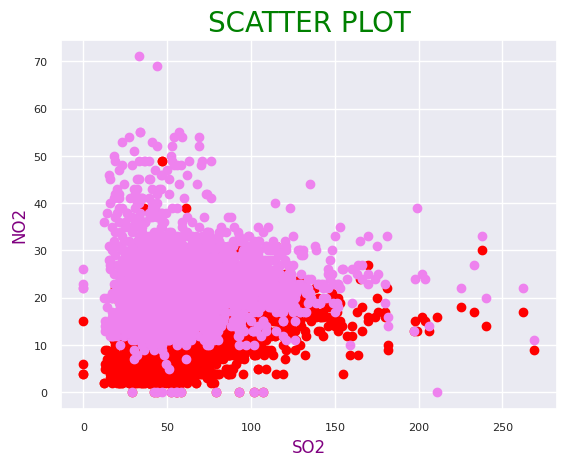
**plt.ylabel("NO2",color="purple")**

**plt.tick\_params(axis='x', which='major', labelsize=8)**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

**plt.show()**

**Output:**

****

**Input:**

**x=df['City/Town/Village/Area']**

**y=df['SO2']**

**plt.bar(x,y,color="indigo")**

**plt.title("BAR PLOT",color="Green",fontsize=20)**

**plt.xlabel("City/Town/Village/Area",color="red")**

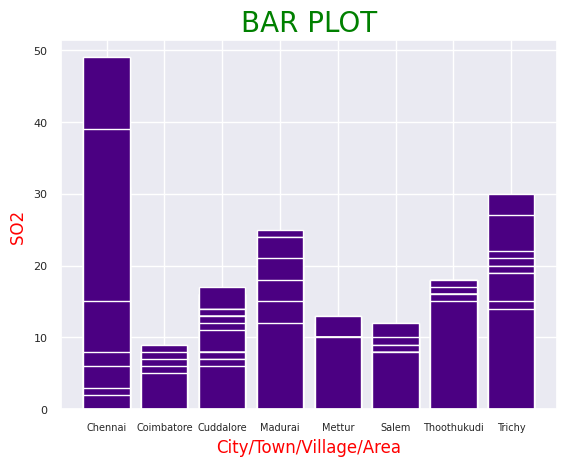
**plt.ylabel("SO2 ",color="red")**

**plt.tick\_params(axis='x', which='major', labelsize=7)**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

**plt.show()**

**Output:**

****

**Input:**

**y=df['NO2']**

**plt.hist(y,bins=15)**

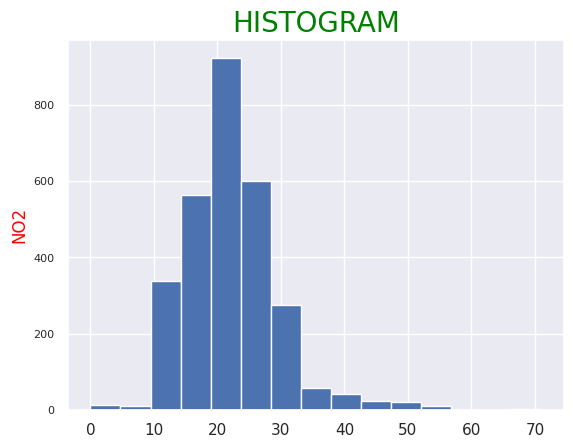
**plt.title("HISTOGRAM",color="Green",fontsize=20)**

**plt.ylabel("NO2",color="red")**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

**plt.show()**

**Output:**

****

**Input:**

**x=df['SO2'].value\_counts()**

**y=['11.',' 13.',' 12.',' 15.',' 14.',' 10.',' 16.',' 19.',' 9.',' 20.', '17.',' 18.',' 25.', '21.',' 23.',' 26.',' 24.', '32.',' 27.',' 30.',' 22.',' 0.',' 8.',' 31.',' 28.',' 29.','6.',' 49.',' 3.',' 7.',' 5.',' 2.',' 4.',' 39.']**

**plt.pie(x.values,labels=y)**

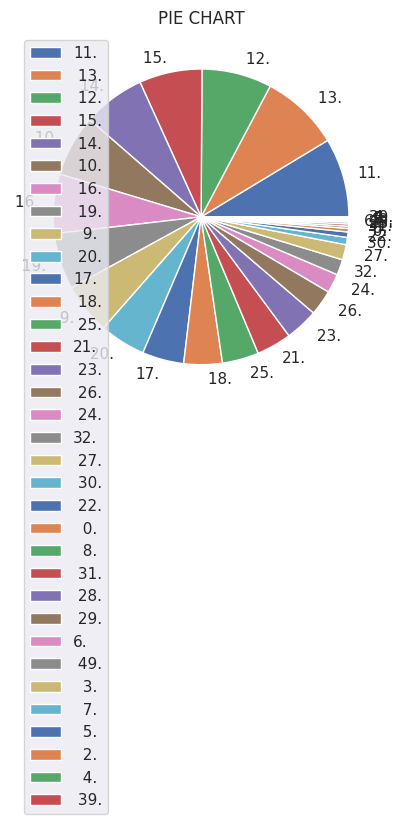
**plt.title('PIE CHART')**

**plt.tick\_params(axis='x', which='major', labelsize=8)**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

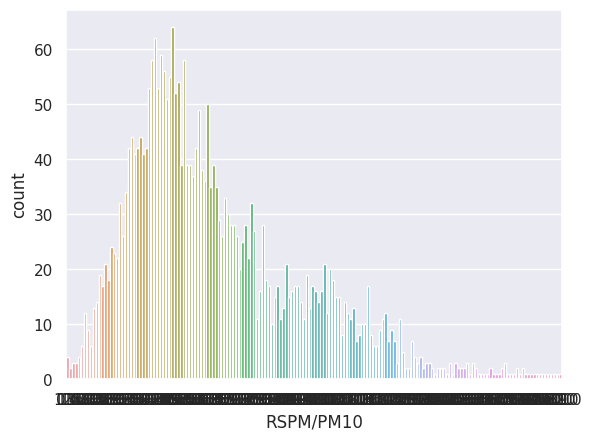
**plt.legend()**

**Output:**

****

**Input: sns.countplot(x='RSPM/PM10',data=df)**

**Output:**

****

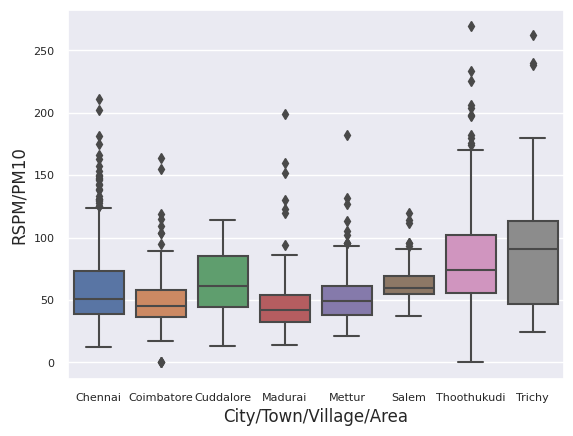
**Input:**

**sns.boxplot(x='City/Town/Village/Area',y='RSPM/PM10',data=df)**

**plt.tick\_params(axis='x', which='major', labelsize=8)**

**plt.tick\_params(axis='y', which='major', labelsize=8)**

**Output:**

****

**Step 7: Calculate Average Pollution Levels**

Calculate the average levels of SO2, NO2, and RSPM/PM10 across different monitoring stations, cities, or areas. You can use the Pandas groupby function to group the data by the relevant columns (e.g., station name, city), and then calculate the means.

**Input:**

**# Group by monitoring station and calculate average levels**

**avg\_so2 = df.groupby('Location of Monitoring Station')['SO2'].mean()**

**avg\_no2 = df.groupby('Location of Monitoring Station')['NO2'].mean()**

**avg\_rspm\_pm10 = df.groupby('Location of Monitoring Station')['RSPM/PM10'].mean()**

**Step 8: Identify Pollution Trends:**

Analyze the calculated averages to identify pollution trends and areas with high pollution levels. You can sort and filter the data to find the highest and lowest pollution levels.

**Input:**

**# Find the station with the highest average SO2 level**

**highest\_so2\_station = avg\_so2.idxmax()**

**highest\_so2\_value = avg\_so2.max()**

**# Find the station with the highest average NO2 level**

**highest\_no2\_station = avg\_no2.idxmax()**

**highest\_no2\_value = avg\_no2.max()**

**# Find the station with the highest average RSPM/PM10 level**

**highest\_rspm\_pm10\_station = avg\_rspm\_pm10.idxmax()**

**highest\_rspm\_pm10\_value = avg\_rspm\_pm10.max()**

**Step 9: Create Visualizations**

Use data visualization libraries such as Matplotlib and Seaborn to create visualizations that help convey your findings.To create bar plots to visualize the average pollution levels:

**Input:**

**# Create bar plots for average pollution levels**

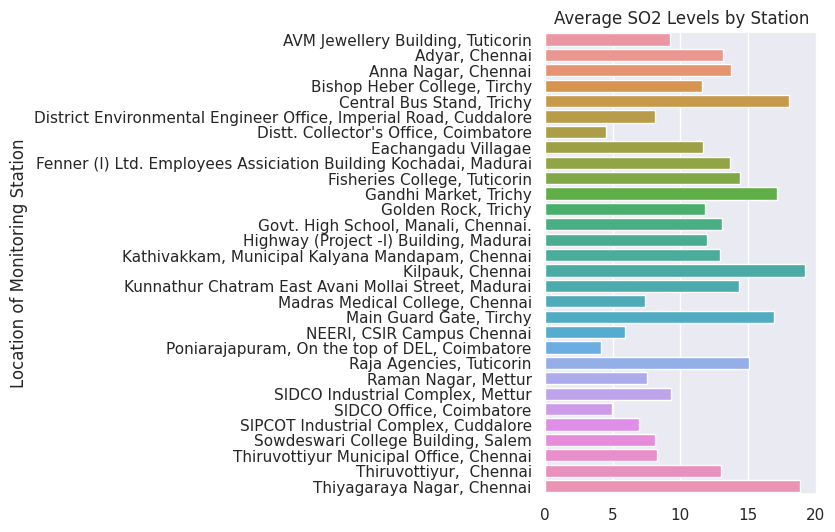
**plt.figure(figsize=(12, 6))**

**plt.subplot(131)**

**sns.barplot(x=avg\_so2.values, y=avg\_so2.index)**

**plt.title('Average SO2 Levels by Station')**

**Output:**

****

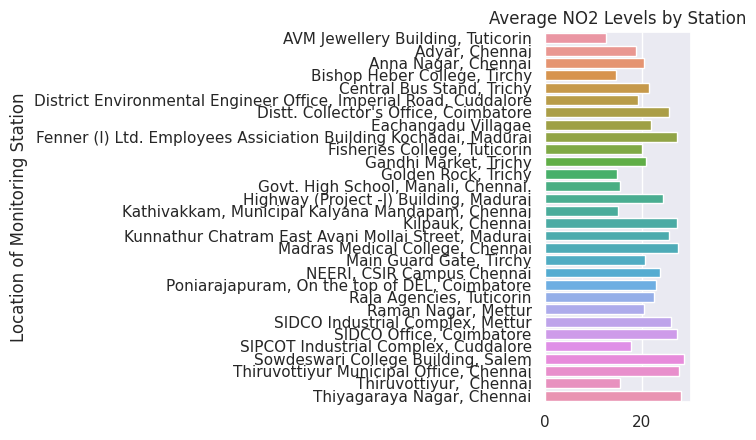
**Input:**

**plt.subplot(132)**

**sns.barplot(x=avg\_no2.values, y=avg\_no2.index)**

**plt.title('Average NO2 Levels by Station')**

**Output:**

****

**Input:**

**plt.subplot(133)**

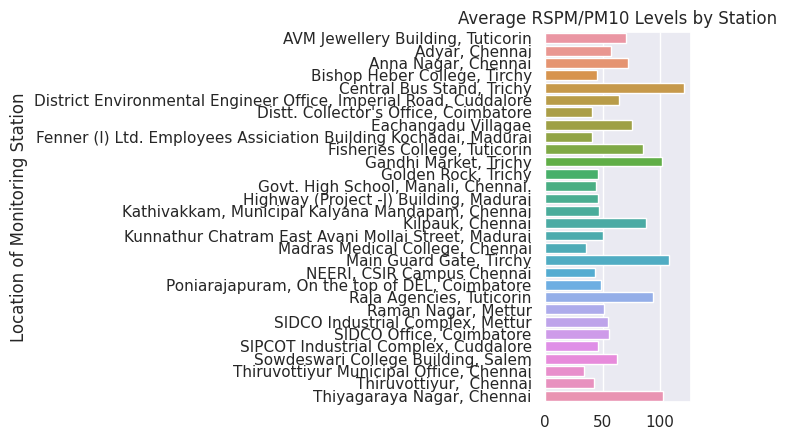
**sns.barplot(x=avg\_rspm\_pm10.values, y=avg\_rspm\_pm10.index)**

**plt.title('Average RSPM/PM10 Levels by Station')**

**plt.tight\_layout()**

**plt.show()**

**Output:**

****

**Step 10: Visualization based on City/Town/Village/Area and calculate average levels**

**Input:**

**# Group by City/Town/Village/Area and calculate average levels**

**avg\_so2 = df.groupby('City/Town/Village/Area')['SO2'].mean()**

**avg\_no2 = df.groupby('City/Town/Village/Area')['NO2'].mean()**

**avg\_rspm\_pm10 = df.groupby('City/Town/Village/Area')['RSPM/PM10'].mean()**

**Step 11: Identify Pollution Trends**

Analyze the calculated averages to identify pollution trends and areas with high pollution levels. Here's how you can find the stations with the highest average levels:

**Input:**

**# Find the City/Town/Village/Area with the highest average SO2 level**

**highest\_so2\_station = avg\_so2.idxmax()**

**highest\_so2\_value = avg\_so2.max()**

**# Find the City/Town/Village/Area with the highest average NO2 level**

**highest\_no2\_station = avg\_no2.idxmax()**

**highest\_no2\_value = avg\_no2.max()**

**# Find the City/Town/Village/Area with the highest average RSPM/PM10 level**

**highest\_rspm\_pm10\_station = avg\_rspm\_pm10.idxmax()**

**highest\_rspm\_pm10\_value = avg\_rspm\_pm10.max()**

**Input:**

**# Create bar plots for average pollution levels**

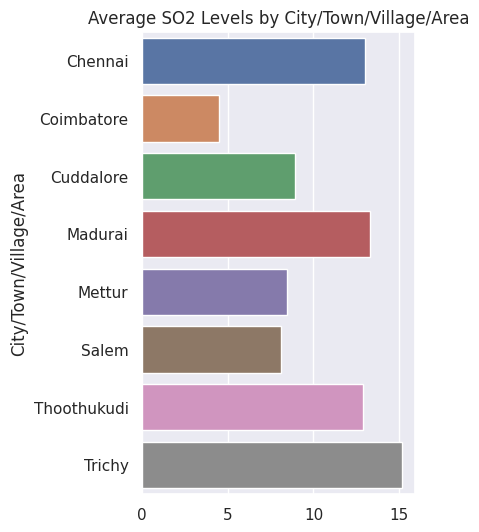
**plt.figure(figsize=(12, 6))**

**plt.subplot(131)**

**sns.barplot(x=avg\_so2.values, y=avg\_so2.index)**

**plt.title('Average SO2 Levels by City/Town/Village/Area')**

**Output:**

****

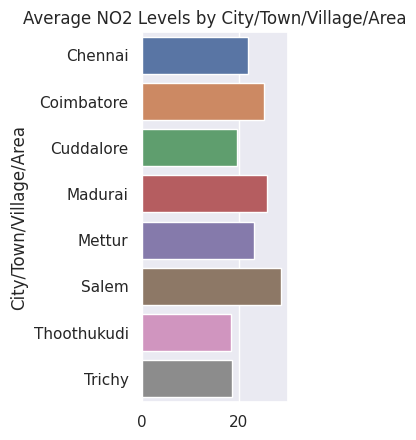
**Input:**

**plt.subplot(132)**

**sns.barplot(x=avg\_no2.values, y=avg\_no2.index)**

**plt.title('Average NO2 Levels by City/Town/Village/Area')**

**Output:**

****

**Input:**

**plt.subplot(133)**

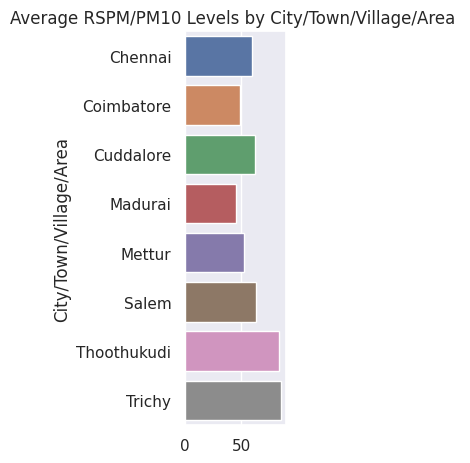
**sns.barplot(x=avg\_rspm\_pm10.values, y=avg\_rspm\_pm10.index)**

**plt.title('Average RSPM/PM10 Levels by City/Town/Village/Area ')**

**plt.tight\_layout()**

**plt.show()**

**Output:**

****

**Step 11: Save the Visualizations:**

If you want to save the visualizations as image files, you can use plt.savefig('visualization.png') after creating the plot.

**Step 12: Working with Cognos**

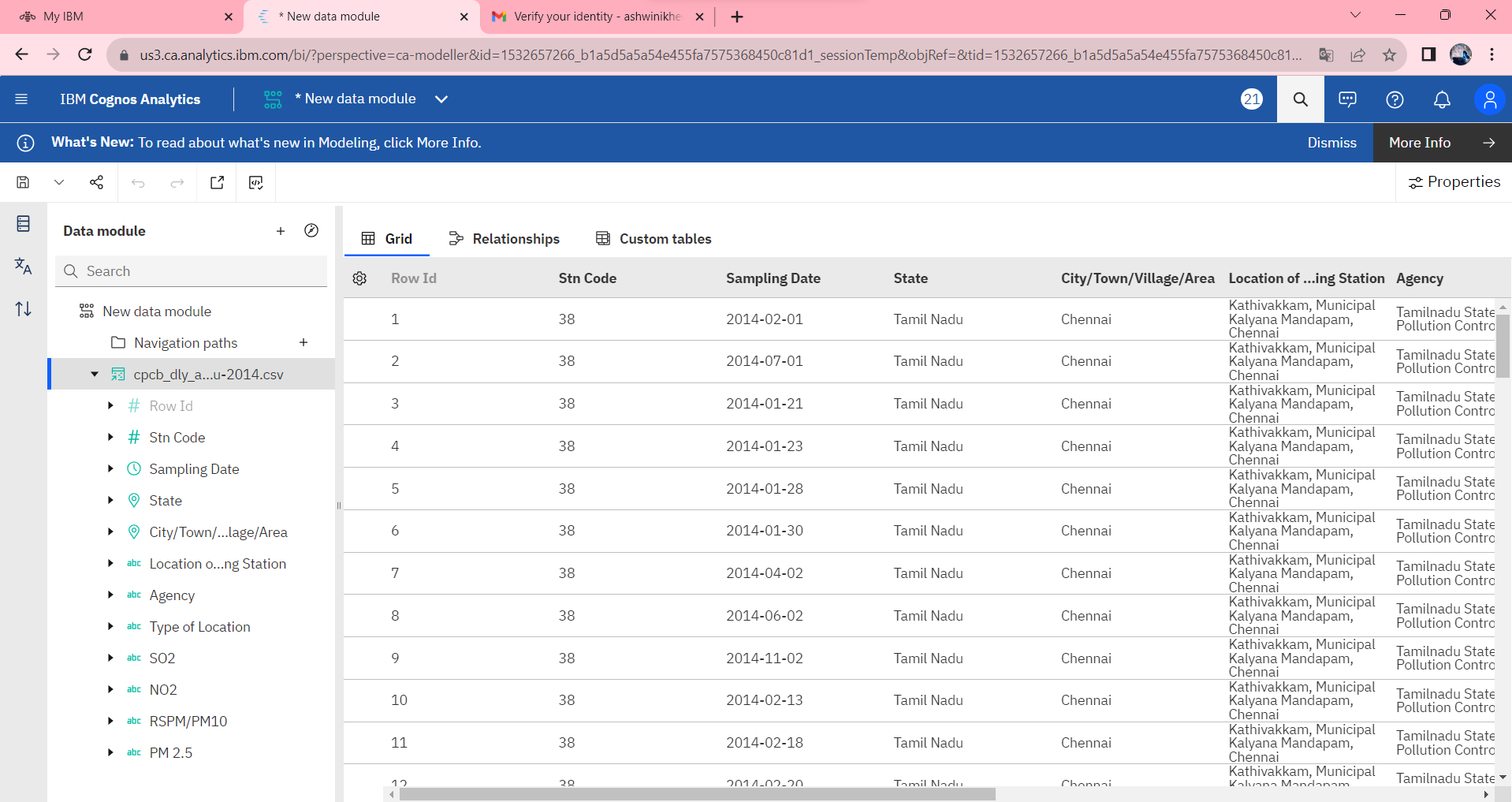
****

Figure 1: No2 by City/Town /village/area

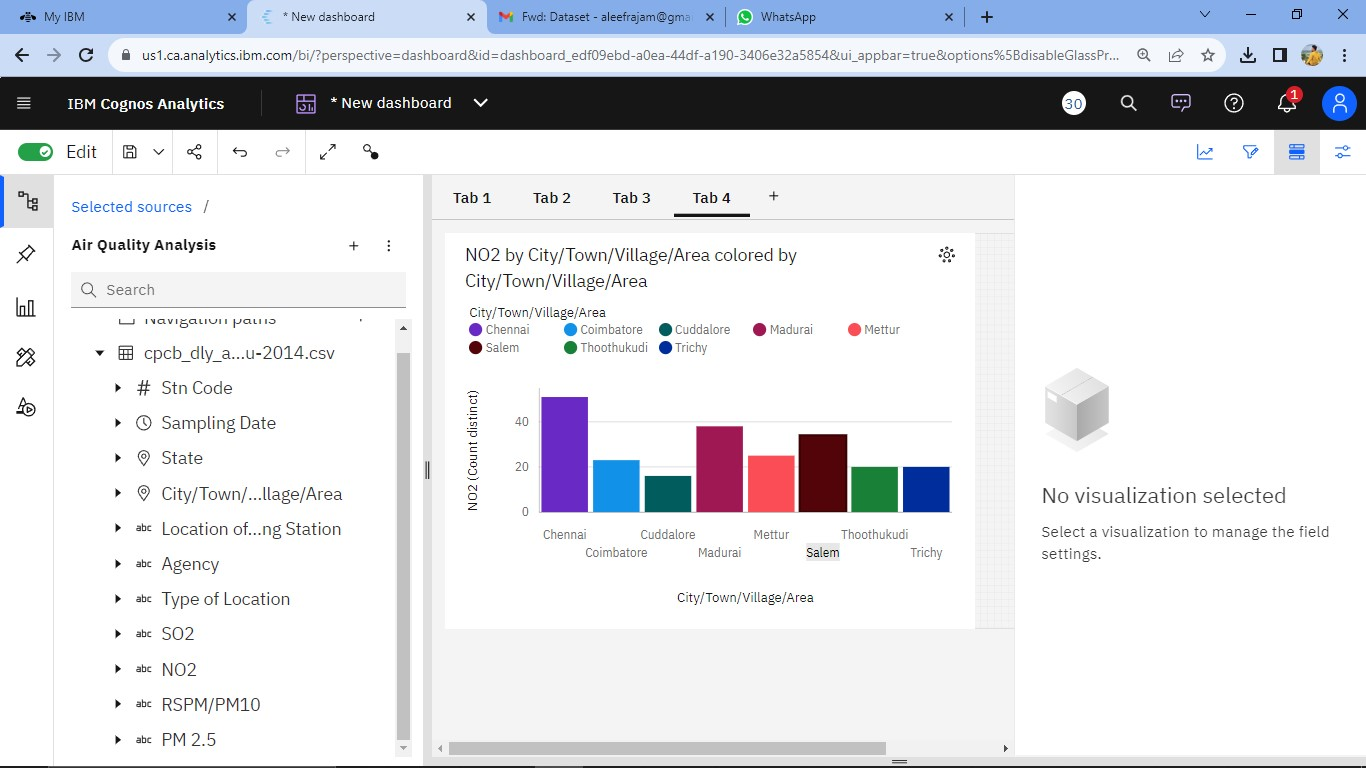


Figure 2: SO2 by City/Town/Village/Area

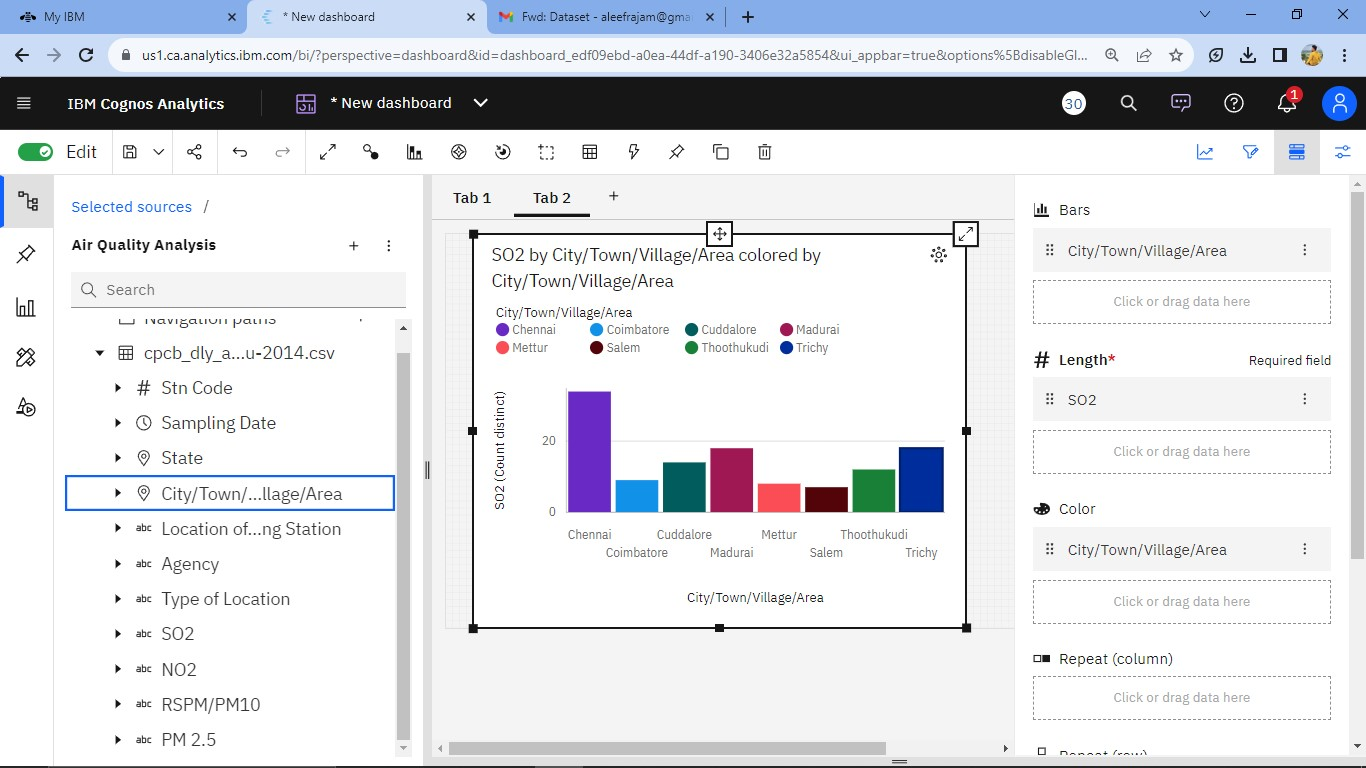


Figure 3: City/Town/Village/Area by NO2 and SO2

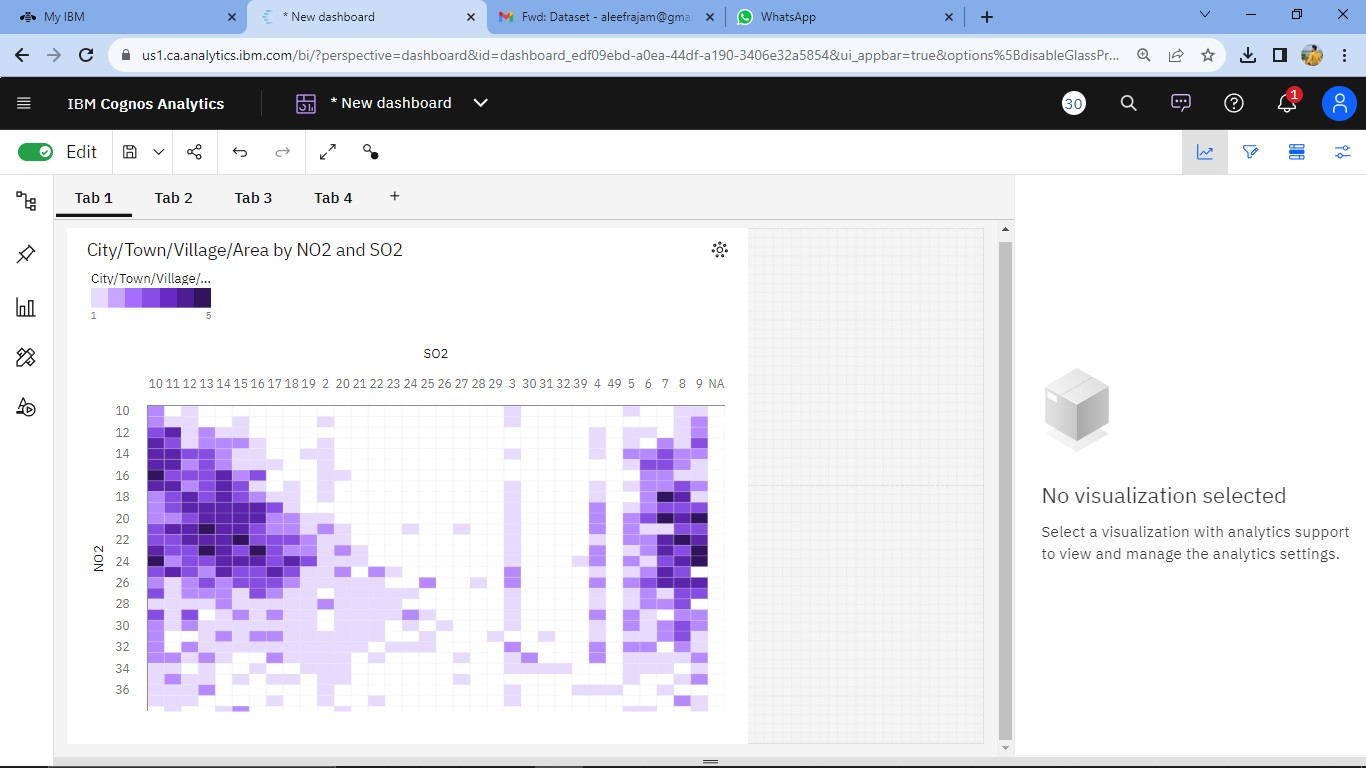
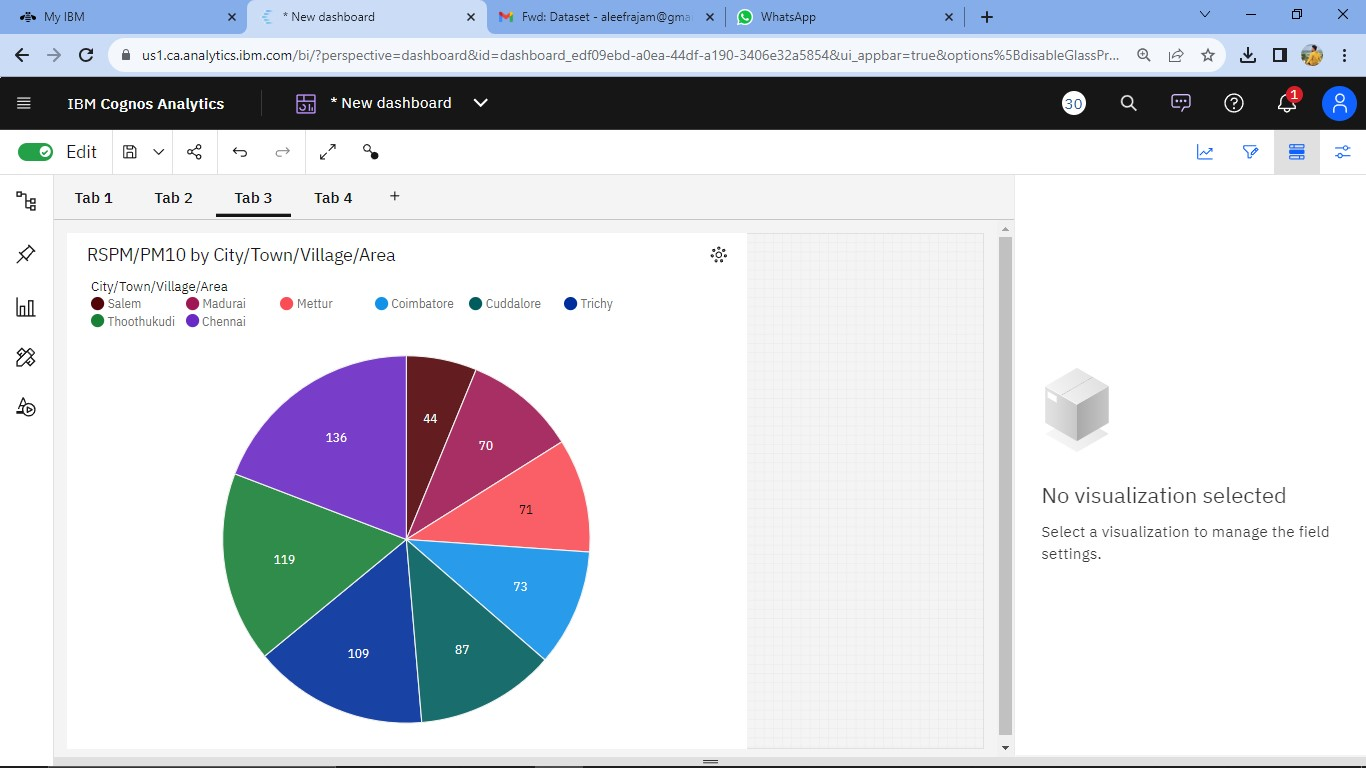


Figure 4: RSPM/PM10 by City/Town/Village/Area



**GitHub Repository Link:**

Provided the link to our GitHub repository containing the project's code and dataset.

Abinaya M- [github repository link](https://github.com/AbinayaMurugesa/DAC_PROJECT.git)

Alef Raja M -[github repository link](https://github.com/ALEEFRAJA/Problem-statement-and-design-thinking.git)

Ashwini S - [github repository link](https://github.com/AshwiniKheersagar/problem-statement-and-design-thinking.git)

Jayanthi E -[github repository link](https://github.com/Jaihai/IBM-PROJECT.git)

Kiruthiga K -[github repository link](https://github.com/K-Kiruthiga/Phase-2.git)

**Replicating the Analysis:**

To replicate the analysis, follow these steps:

* Download the dataset from the provided link.
* Install Python and the Pandas library if not already installed: pip install pandas.
* Load the dataset into a Pandas DataFrame using the provided code.
* Preprocess the data as needed.
* Follow the code to create visualizations and calculate average pollution levels.
* Analyze the findings to gain insights into air pollution trends and pollution levels.

**Key Findings Of Project:**

The code provides a visual overview of the data and initial insights into air quality in Tamil Nadu in 2014.

The highest average pollution levels and corresponding monitoring stations or areas are identified for SO2, NO2, and RSPM/PM10.