Air Quality Analysis in Tamil Nadu

Phase - 5

1. **Project Objectives:**

#The project’s objective was to analyze air quality in Tamil Nadu by examining historical data, with a focus on SO2, NO2, and RSPM/PM10 levels. We implemented data preprocessing, visualization, and linear regression modeling. Example outputs include time series plots and predictive models.

#This analysis offers insights into trends, showing fluctuations in pollutant levels over time. The code aids in estimating air quality, providing valuable information for understanding and managing pollution in Tamil Nadu.

1. **ANALYSIS**
2. **The analysis of air quality data in Tamil Nadu offers valuable insights into air pollution trends and pollution levels in the region in the following ways:**

#**Temporal Trends:** #By visualizing time series data for pollutants like SO2, NO2, and RSPM/PM10, the analysis reveals how pollution levels change over time. Patterns, fluctuations, and long-term trends become apparent, aiding in understanding the impact of various factors on air quality.

#**Spatial Variations:** #Heatmaps and location-based visualizations show variations in pollutant levels across different monitoring stations in Tamil Nadu. This helps identify areas with consistently high or low pollution, which can inform policy decisions and resource allocation.

#**Statistical Summaries:** #Summary statistics and box plots provide a comprehensive view of the distribution of pollutant concentrations. Mean, median, and standard deviation values, as well as potential outliers, offer insights into the central tendency and variability of air pollution levels.

#**Regression Modeling:** #The linear regression model for estimating RSPM/PM10 levels based on SO2 and NO2 concentrations allows us to make predictions and understand how these key pollutants contribute to particulate matter in the air. This can be crucial for pollution control efforts.

#Overall, the analysis enables stakeholders, policymakers, and researchers to gain a deeper understanding of air pollution trends in Tamil Nadu. This understanding can drive informed decisions and interventions to improve air quality, protect public health, and mitigate environmental impacts.

[ ]:

**import**

**pandas**

**as**

**pd**

**import**

**matplotlib**

**.**

**pyplot**

**as**

**plt**

[ ]:

*# Assuming your data is in a CSV file*

data

=

pd

.

read\_csv(

'

/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv

'

)

1. **Data Preprocessing:**

# We cleaned and structured the air quality dataset, handling missing values and data format conversions.

[ ]:

*# Display basic statistics*

print

(

data

.

describe())

*# Check for missing values*

print

(

data

.

isnull()

.

sum())

*# Check unique values in categorical columns*

print

(

data

[

'

State

'

]

.

unique())

print

(

data

[

'

City/Town/Village/Area

'

]

.

unique())

*# ... Repeat for other categorical columns*

Stn Code

SO2

NO2

RSPM/PM10

PM 2.5

|  |  |  |  |
| --- | --- | --- | --- |
| count 2879.000000 2868.000000 | 2866.000000 | 2875.000000 | 0.0 |
| mean 475.750261 11.503138 | 22.136776 | 62.494261 | NaN |
| std 277.675577 5.051702 | 7.128694 | 31.368745 | NaN |
| min 38.000000 2.000000 | 5.000000 | 12.000000 | NaN |
| 25% 238.000000 8.000000 | 17.000000 | 41.000000 | NaN |
| 50% 366.000000 12.000000 | 22.000000 | 55.000000 | NaN |
| 75% 764.000000 15.000000 | 25.000000 | 78.000000 | NaN |
| max 773.000000 49.000000 | 71.000000 | 269.000000 | NaN |
| Stn Code | 0 |  |  |
| Sampling Date | 0 |  |  |
| State | 0 |  |  |
| City/Town/Village/Area | 0 |  |  |
| Location of Monitoring Station | 0 |  |  |
| Agency | 0 |  |  |
| Type of Location | 0 |  |  |
| SO2 | 11 |  |  |
| NO2 | 13 |  |  |
| RSPM/PM10 | 4 |  |  |
| PM 2.5 | 2879 |  |  |

dtype: int64 ['Tamil Nadu']

['Chennai' 'Coimbatore' 'Cuddalore' 'Madurai' 'Mettur' 'Salem' 'Thoothukudi' 'Trichy']

[ ]:

**import**

**matplotlib**

**.**

**pyplot**

**as**

**plt**

*# Plot a line chart for RSPM/PM10 levels over time*

plt

.

figure(figsize

=

(

22

,

12

))

plt

.

plot(data

.

index, data[

'

RSPM/PM10

'

]

, marker

=

'

o

'

, linestyle

=

'

-

'

, color

=

'

b

'

,

␣

↪

label

=

'

RSPM/PM10

'

)

plt

.

xlabel(

'

Date

'

)

plt

.

ylabel(

'

RSPM/PM10 Levels

'

)

plt

.

title(

'

Trends in RSPM/PM10 Pollution Levels Over Time

'

)

plt

.

grid(

**True**

)

plt

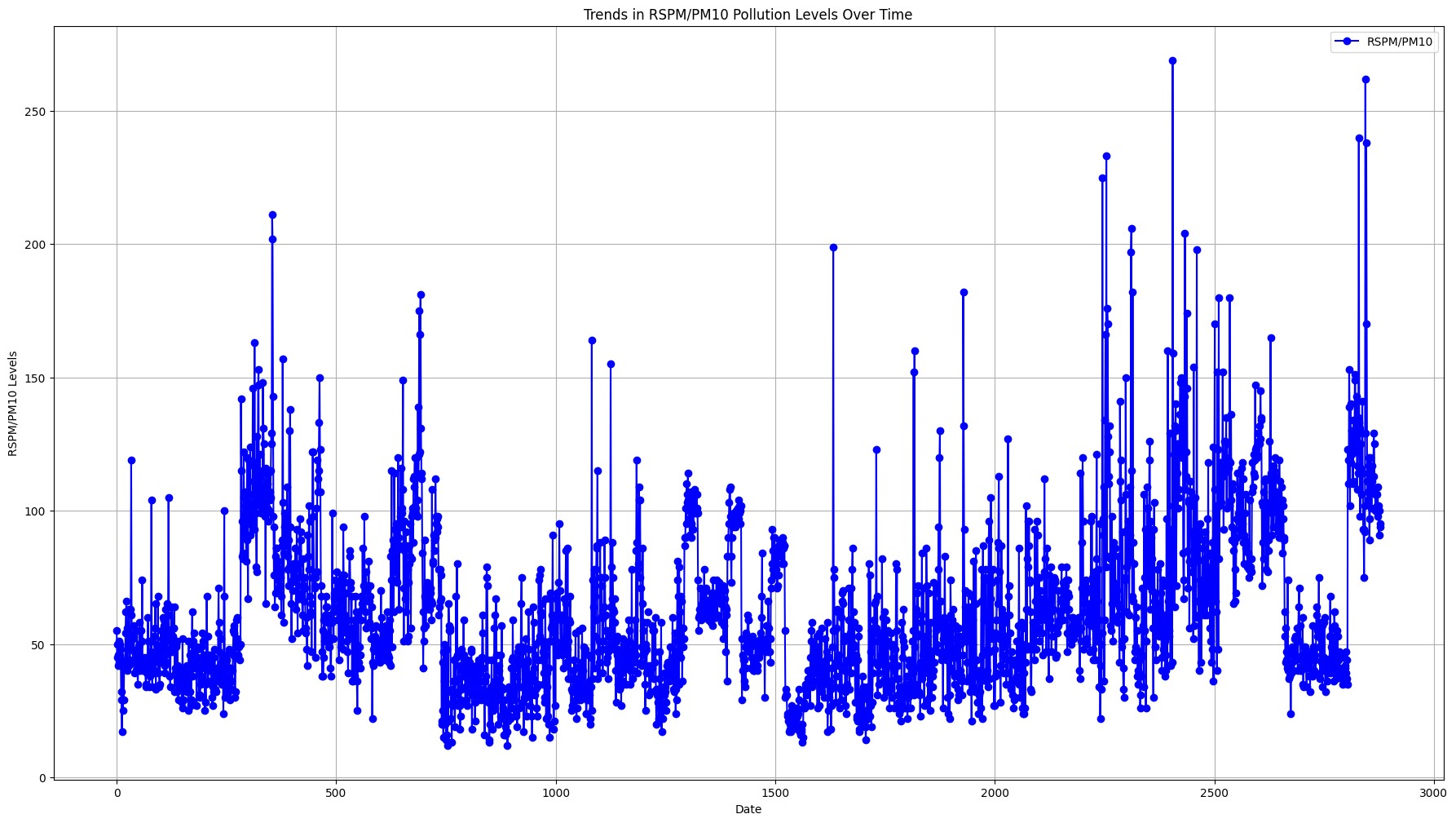
.

legend()

plt

.

show()



1. **Groups the air quality data by date**
2. **Calculates the mean values for SO2, NO2, and RSPM/PM10, and plots the daily average air quality in Tamil Nadu. The line chart provides a clear visualization of how these pollutant concentrations vary over time, aiding in understanding daily air quality trends and fluctuations.**

[ ]:

*# Group data by date and calculate mean values*

daily\_mean

=

data

.

groupby(

'

Sampling Date

'

)

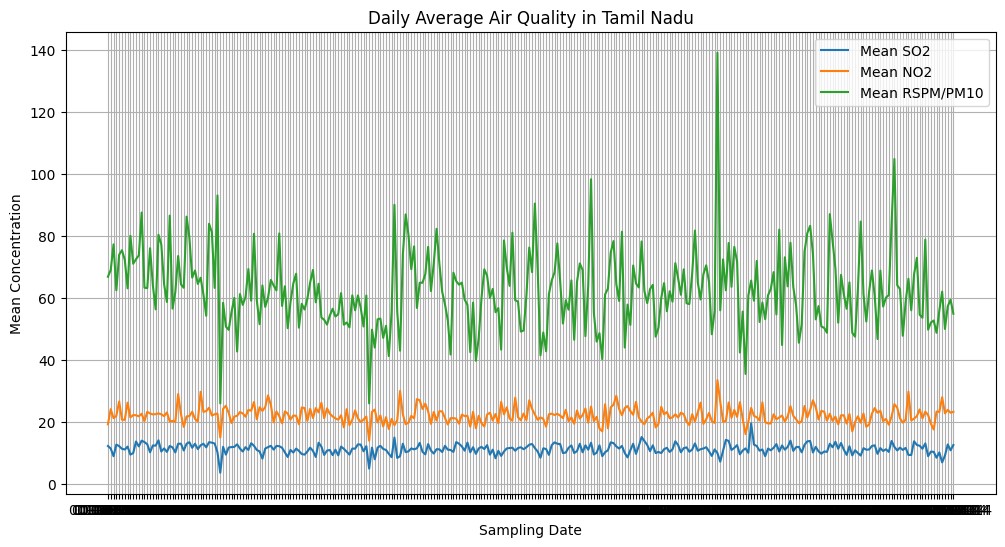
.

mean()

*# Plot daily average air quality*

plt.figure(figsize=(12, 6)) plt.plot(daily\_mean.index, daily\_mean['SO2'], label='Mean SO2') plt.plot(daily\_mean.index, daily\_mean['NO2'], label='Mean NO2') plt.plot(daily\_mean.index, daily\_mean['RSPM/PM10'], label='Mean RSPM/PM10') plt.xlabel('Sampling Date') plt.ylabel('Mean Concentration') plt.title('Daily Average Air Quality in Tamil Nadu') plt.legend() plt.grid(**True**) plt.show()

<ipython-input-5-4b27baf3318f>:2: 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. daily\_mean = data.groupby('Sampling Date').mean()



1. **Concentrations of Sulfur Dioxide (SO2) and Nitrogen Dioxide (NO2)**

#Calculates the daily average concentrations of Sulfur Dioxide (SO2) and Nitrogen Dioxide (NO2) in Tamil Nadu. The resulting line chart provides insights into how these pollutants’ levels change over time. It aids in visualizing and understanding the variations in SO2 and NO2 concentrations, essential for monitoring air quality in the region.

[ ]:

*# Calculate daily average SO2 and NO2 concentrations for all monitoring stations*

daily\_mean

=

data

.

groupby(

'

Sampling Date

'

)[[

'

SO2

'

,

'

NO2

'

]]

.

mean()

*# Plot daily average SO2 and NO2 concentrations*

plt

.

figure(figsize

=

(

12

,

6

))

plt

.

plot(daily\_mean

.

index, daily\_mean[

'

SO2

'

]

, label

=

'

Mean SO2 Concentration

'

)

plt

.

plot(daily\_mean

.

index, daily\_mean[

'

NO2

'

]

, label

=

'

Mean NO2 Concentration

'

)

plt

.

xlabel(

'

Sampling Date

'

)

plt

.

ylabel(

'

Mean Concentration (µg/m³)

'

)

*# Units may vary based on your data*

plt

.

title(

'

Daily Average SO2 and NO2 Concentrations in Tamil Nadu

'

)

plt

.

legend()

plt

.

grid(

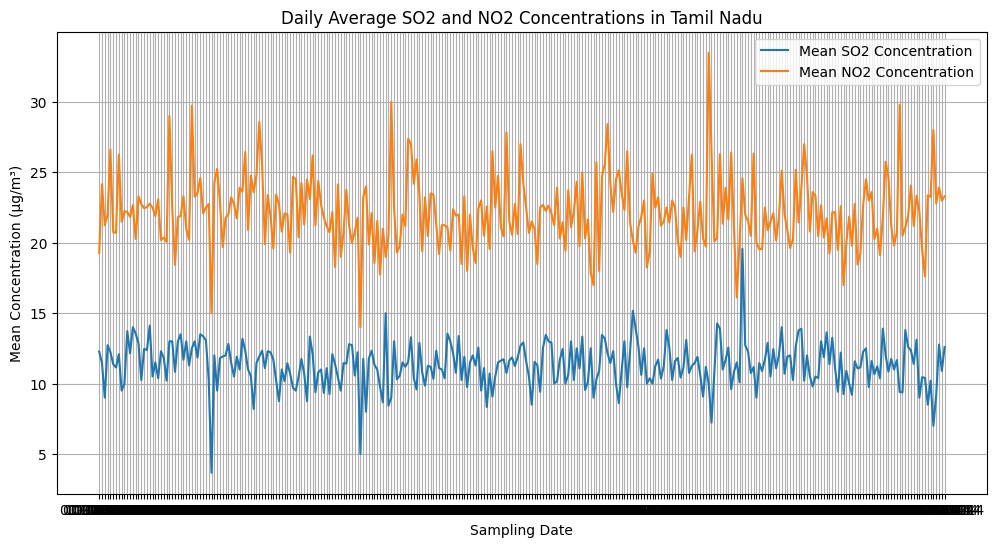
**True**

)

plt

.

show()



1. **Displays statistics for Sulfur Dioxide (SO2) and Nitrogen Dioxide (NO2) concentrations**

# computes and displays statistics for Sulfur Dioxide (SO2) and Nitrogen Dioxide (NO2) concentrations. It generates two box plots, one for SO2 and the other for NO2. These plots help visualize the distribution of concentration values, showing the central tendency, spread, and potential outliers in the data, aiding in the identification of extreme values and data variability.

[ ]:

*# Summary statistics*

so2\_stats

=

data[

'

SO2

'

]

.

describe()

no2\_stats

=

data[

'

NO2

'

]

.

describe()

*# Box plots to visualize the distribution and identify outliers*

plt

.

figure(figsize

=

(

12

,

6

))

plt

.

subplot(

1

,

2

,

1

)

data

.

boxplot(column

=

'

SO2

'

)

plt

.

title(

'

SO2 Concentration Box Plot

'

)

plt

.

subplot(

1

,

2

,

2

)

data

.

boxplot(column

=

'

NO2

'

)

plt

.

title(

'

NO2 Concentration Box Plot

'

)

plt

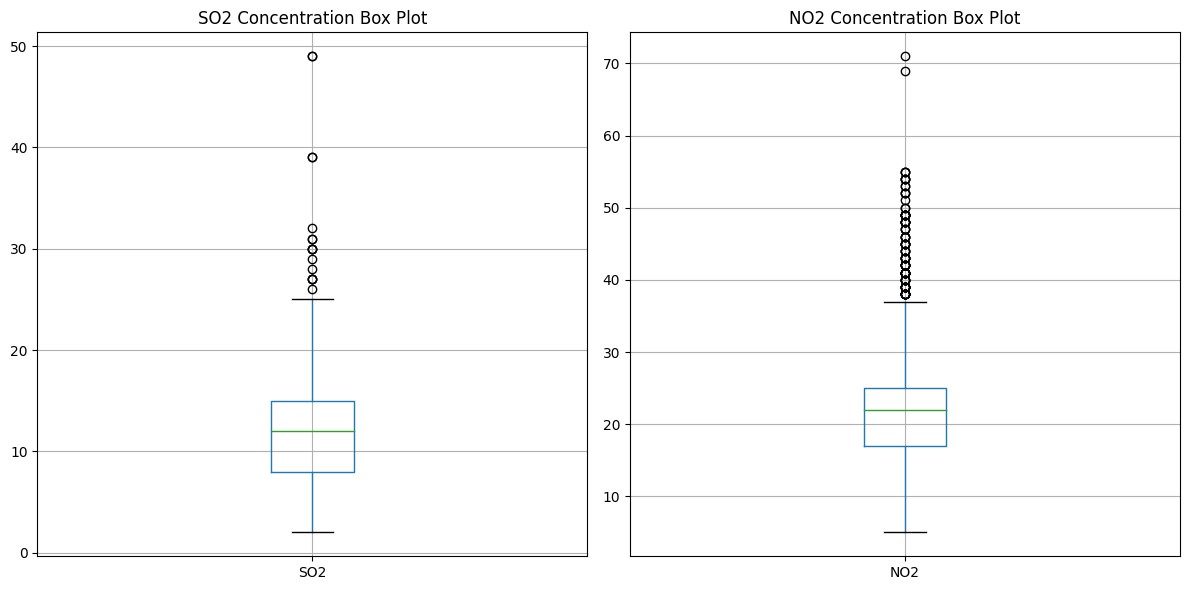
.

tight\_layout()

plt

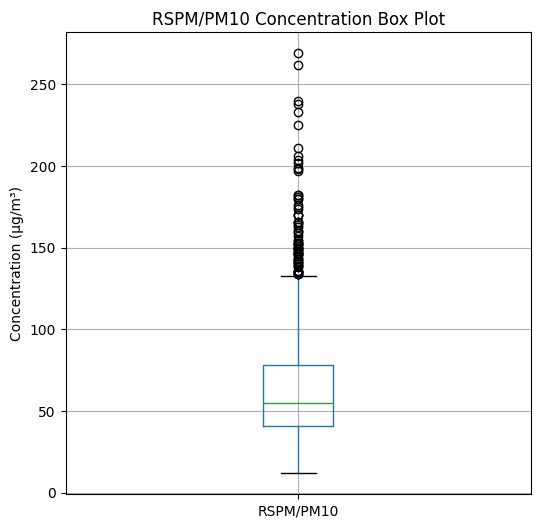
.

show()



[ ]: *# Summary statistics* rspm\_pm10\_stats = data['RSPM/PM10'].describe()

*# Box plot to visualize the distribution and identify outliers* plt.figure(figsize=(6, 6)) data.boxplot(column='RSPM/PM10') plt.title('RSPM/PM10 Concentration Box Plot') plt.ylabel('Concentration (µg/m³)') *# Units may vary based on your data* plt.grid(**True**) plt.show()



[ ]:

**import**

**seaborn**

**as**

**sns**

*# Preprocess the data*

*# - Convert the 'Sampling Date' column to datetime format*

data[

'

Sampling Date

'

]

=

pd

.

to\_datetime(data[

'

Sampling Date

'

])

*# - Filter data for Tamil Nadu*

tn\_data

=

data[data[

'

State

'

]

==

'

Tamil Nadu

'

]

*# Visualize trends in air pollution over time*

plt

.

figure(figsize

=

(

12

,

6

))

*# Plot time series of SO2, NO2, and RSPM/PM10 concentrations*

**for**

pollutant

**in**

[

'

SO2

'

,

'

NO2

'

,

'

RSPM/PM10

'

]:

sns

.

lineplot(data

=

tn\_data, x

=

'

Sampling Date

'

,y

=

pollutant, label

=

pollutant)

plt

.

xlabel(

'

Year

'

)

plt

.

ylabel(

'

Concentration (µg/m³)

'

)

plt

.

title(

'

Air Pollution Trends in Tamil Nadu

'

)

plt

.

legend()

plt

.

grid(

**True**

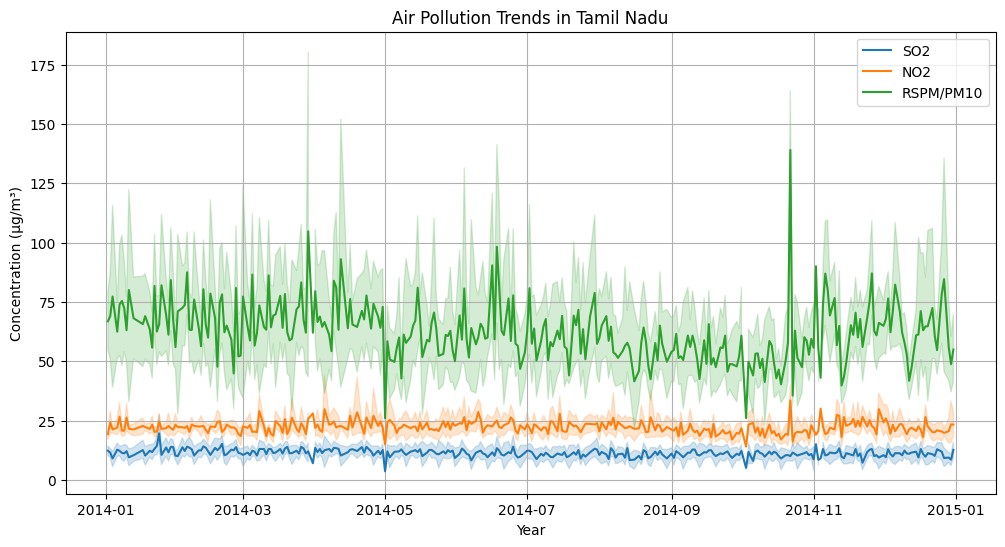
)

plt

.

show()

<ipython-input-9-8fdd52923255>:5: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consistent parsing. data['Sampling Date'] = pd.to\_datetime(data['Sampling Date'])



1. **Data Visualization:**
2. **We used Matplotlib and Seaborn to create visualizations such as time series plots to explore pollutant trends.**

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

avg\_so2\_no2\_by\_location = data.groupby('City/Town/Village/Area')[['SO2',␣ ↪'NO2']].mean().reset\_index()

*# Create a bar chart for average SO2 levels by location* plt.figure(figsize=(12, 6)) plt.bar(avg\_so2\_no2\_by\_location['City/Town/Village/Area'],␣

↪avg\_so2\_no2\_by\_location['SO2'], label='Average SO2 Levels', alpha=0.7,␣ ↪color='b')

*# Create a bar chart for average NO2 levels by location*

plt

.

bar(avg\_so2\_no2\_by\_location[

'

City/Town/Village/Area

'

]

,

␣

↪

avg\_so2\_no2\_by\_location[

'

NO2

'

]

, label

=

'

Average NO2 Levels

'

, alpha

=

0.7

,

␣

↪

color

=

'

b

'

)

plt

.

xlabel(

'

Location

'

)

plt

.

ylabel(

'

Average Levels

'

)

plt

.

title(

'

Average SO2 and NO2 Levels by Location

'

)

plt

.

xticks(rotation

=

90

)

*# Rotate x-axis labels for better readability*

plt

.

legend()

plt

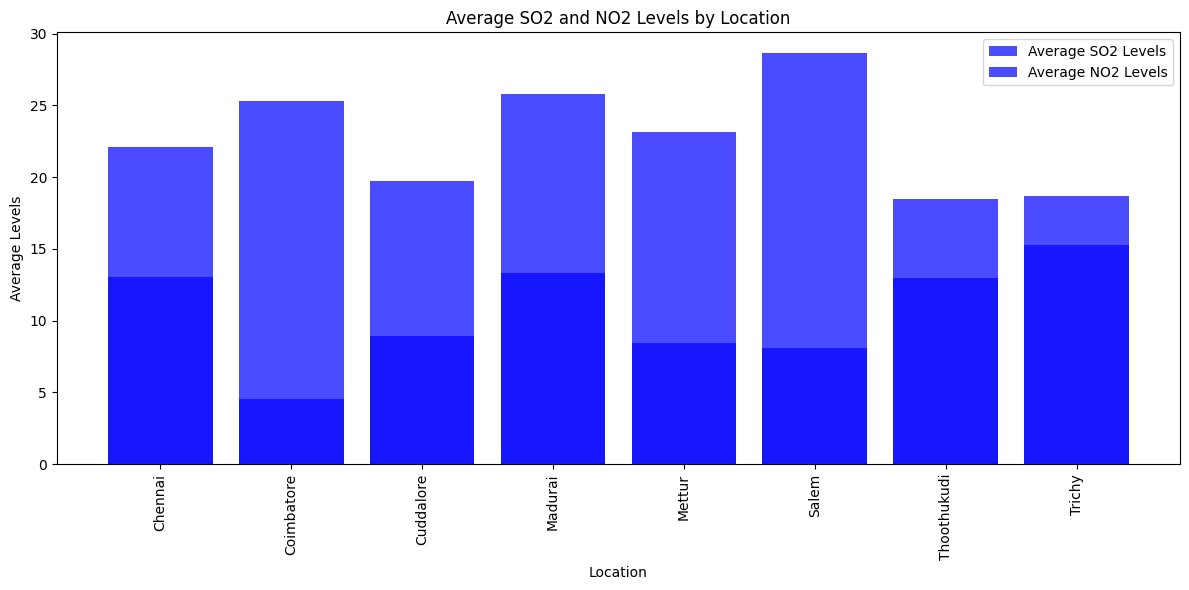
.

tight\_layout()

plt

.

show()



[ ]: **from sklearn.model\_selection import** train\_test\_split **from sklearn.linear\_model import** LinearRegression **from sklearn.metrics import** mean\_squared\_error, r2\_score

*# Preprocess the data and select relevant columns* data['Sampling Date'] = pd.to\_datetime(data['Sampling Date']) tn\_data = data[data['State'] == 'Tamil Nadu'] selected\_columns = ['SO2', 'NO2', 'RSPM/PM10']

tn\_data = tn\_data[selected\_columns].dropna() *# Remove rows with missing values*

*# Split the data into training and testing sets*

X = tn\_data[['SO2', 'NO2']] y = tn\_data['RSPM/PM10']

X\_train, X\_test, y\_train, y\_test

=

train\_test\_split(X, y, test\_size

=

0.2

,

␣

↪

random\_state

=

42

)

*# Create and train a linear regression model*

model

=

LinearRegression()

model

.

fit(X\_train, y\_train)

*# Make predictions on the test set*

y\_pred

=

model

.

predict(X\_test)

*# Evaluate the model*

mse

=

mean\_squared\_error(y\_test, y\_pred)

r2

=

r2\_score(y\_test, y\_pred)

print

(

f

'

Mean Squared Error:

**{**

mse

**}**

'

)

print

(

f

'

R-squared (Coefficient of Determination):

**{**

r2

**}**

'

)

*# Plot the predicted vs. actual values*

plt

.

figure(figsize

=

(

8

,

8

))

plt

.

scatter(y\_test, y\_pred)

plt

.

xlabel(

'

Actual RSPM/PM10 Levels

'

)

plt

.

ylabel(

'

Predicted RSPM/PM10 Levels

'

)

plt

.

title(

'

Actual vs. Predicted RSPM/PM10 Levels

'

)

plt

.

grid(

**True**

)

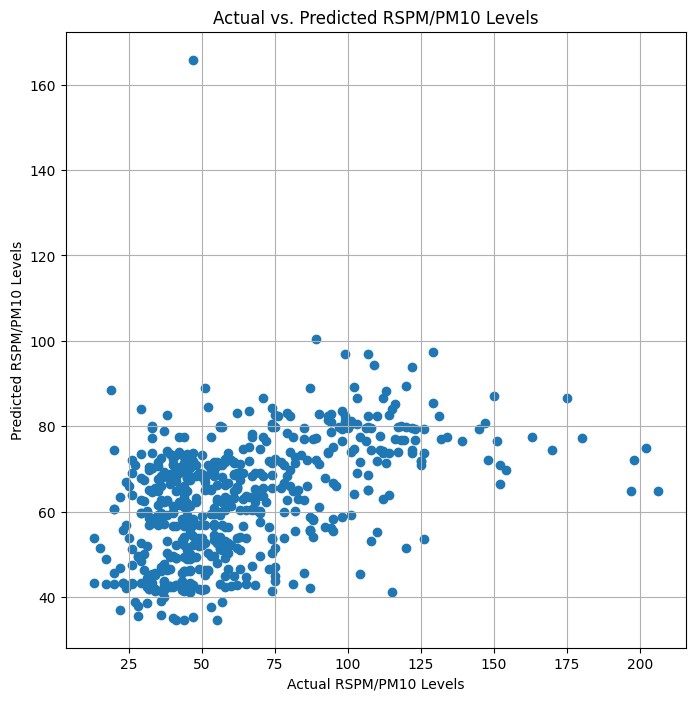
plt

.

show()

Mean Squared Error: 835.4788249190386

R-squared (Coefficient of Determination): 0.20658507746336507



[ ]:

!

pip install folium geopandas

Requirement already satisfied: folium in /usr/local/lib/python3.10/dist-packages

(0.14.0)

Requirement already satisfied: geopandas in /usr/local/lib/python3.10/distpackages (0.13.2)

Requirement already satisfied: branca>=0.6.0 in /usr/local/lib/python3.10/distpackages (from folium) (0.6.0)

Requirement already satisfied: jinja2>=2.9 in /usr/local/lib/python3.10/distpackages (from folium) (3.1.2)

Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages

(from folium) (1.23.5)

Requirement already satisfied: requests in /usr/local/lib/python3.10/distpackages (from folium) (2.31.0)

Requirement already satisfied: fiona>=1.8.19 in /usr/local/lib/python3.10/distpackages (from geopandas) (1.9.5)

Requirement already satisfied: packaging in /usr/local/lib/python3.10/distpackages (from geopandas) (23.2)

Requirement already satisfied: pandas>=1.1.0 in /usr/local/lib/python3.10/distpackages (from geopandas) (1.5.3)

Requirement already satisfied: pyproj>=3.0.1 in /usr/local/lib/python3.10/distpackages (from geopandas) (3.6.1)

Requirement already satisfied: shapely>=1.7.1 in /usr/local/lib/python3.10/distpackages (from geopandas) (2.0.2)

Requirement already satisfied: attrs>=19.2.0 in /usr/local/lib/python3.10/distpackages (from fiona>=1.8.19->geopandas) (23.1.0)

Requirement already satisfied: certifi in /usr/local/lib/python3.10/distpackages (from fiona>=1.8.19->geopandas) (2023.7.22)

Requirement already satisfied: click~=8.0 in /usr/local/lib/python3.10/distpackages (from fiona>=1.8.19->geopandas) (8.1.7) Requirement already satisfied: click-plugins>=1.0 in

/usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (1.1.1) Requirement already satisfied: cligj>=0.5 in /usr/local/lib/python3.10/distpackages (from fiona>=1.8.19->geopandas) (0.7.2)

Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages

(from fiona>=1.8.19->geopandas) (1.16.0)

Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-

packages (from fiona>=1.8.19->geopandas) (67.7.2) Requirement already satisfied: MarkupSafe>=2.0 in

/usr/local/lib/python3.10/dist-packages (from jinja2>=2.9->folium) (2.1.3)

Requirement already satisfied: python-dateutil>=2.8.1 in

/usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2.8.2) Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-

packages (from pandas>=1.1.0->geopandas) (2023.3.post1) Requirement already satisfied: charset-normalizer<4,>=2 in

/usr/local/lib/python3.10/dist-packages (from requests->folium) (3.3.1) Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/distpackages (from requests->folium) (3.4)

Requirement already satisfied: urllib3<3,>=1.21.1 in

/usr/local/lib/python3.10/dist-packages (from requests->folium) (2.0.7)

[ ]:

!

pip install gmaps

Collecting gmaps

Downloading gmaps-0.9.0.tar.gz (1.1 MB)

1.1/1.1 MB

8.2 MB/s eta 0:00:00

Preparing metadata (setup.py) … done

Requirement already satisfied: ipython>=5.3.0 in /usr/local/lib/python3.10/distpackages (from gmaps) (7.34.0)

Requirement already satisfied: ipywidgets>=7.0.0 in /usr/local/lib/python3.10/dist-packages (from gmaps) (7.7.1)

Requirement already satisfied: traitlets>=4.3.0 in

/usr/local/lib/python3.10/dist-packages (from gmaps) (5.7.1)

Collecting geojson>=2.0.0 (from gmaps)

Downloading geojson-3.0.1-py3-none-any.whl (15 kB)

Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages

(from gmaps) (1.16.0)

Requirement already satisfied: setuptools>=18.5 in

/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->gmaps) (67.7.2)

Collecting jedi>=0.16 (from ipython>=5.3.0->gmaps)

Downloading jedi-0.19.1-py2.py3-none-any.whl (1.6 MB)

1.6/1.6 MB

46.0 MB/s eta 0:00:00

Requirement already satisfied: decorator in

/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->gmaps) (4.4.2) Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/distpackages (from ipython>=5.3.0->gmaps) (0.7.5)

Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in

/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->gmaps) (3.0.39) Requirement already satisfied: pygments in /usr/local/lib/python3.10/distpackages (from ipython>=5.3.0->gmaps) (2.16.1)

Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-

packages (from ipython>=5.3.0->gmaps) (0.2.0) Requirement already satisfied: matplotlib-inline in

/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->gmaps) (0.1.6) Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-

packages (from ipython>=5.3.0->gmaps) (4.8.0) Requirement already satisfied: ipykernel>=4.5.1 in

/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.0.0->gmaps) (5.5.6)

Requirement already satisfied: ipython-genutils~=0.2.0 in

/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.0.0->gmaps) (0.2.0)

Requirement already satisfied: widgetsnbextension~=3.6.0 in

/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.0.0->gmaps) (3.6.6)

Requirement already satisfied: jupyterlab-widgets>=1.0.0 in

/usr/local/lib/python3.10/dist-packages (from ipywidgets>=7.0.0->gmaps) (3.0.9) Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/distpackages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->gmaps) (6.1.12)

Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/distpackages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->gmaps) (6.3.2)

Requirement already satisfied: parso<0.9.0,>=0.8.3 in

/usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>=5.3.0->gmaps) (0.8.3)

Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.10/dist-packages (from pexpect>4.3->ipython>=5.3.0->gmaps) (0.7.0)

Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/distpackages (from prompt-

toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=5.3.0->gmaps) (0.2.8) Requirement already satisfied: notebook>=4.4.1 in /usr/local/lib/python3.10/dist-packages (from

widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (6.5.5)

Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (3.1.2)

Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/distpackages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (23.2.1) Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/distpackages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (23.1.0)

Requirement already satisfied: jupyter-core>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (5.4.0) Requirement already satisfied: nbformat in /usr/local/lib/python3.10/distpackages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (5.9.2) Requirement already satisfied: nbconvert>=5 in /usr/local/lib/python3.10/distpackages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (6.5.4)

Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from

notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.5.8)

Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from

notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.8.2)

Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from

notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.17.1)

Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages (from

notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.17.1)

Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.10/dist-packages (from

notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.0.0)

Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.10/dist-packages (from jupyterclient->ipykernel>=4.5.1->ipywidgets>=7.0.0->gmaps) (2.8.2)

Requirement already satisfied: platformdirs>=2.5 in

/usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.6.1->notebook>=4.4

.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (3.11.0)

Requirement already satisfied: jupyter-server>=1.8 in

/usr/local/lib/python3.10/dist-packages (from nbclassic>=0.4.7->notebook>=4.4.1-

>widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.24.0)

Requirement already satisfied: notebook-shim>=0.2.3 in

/usr/local/lib/python3.10/dist-packages (from nbclassic>=0.4.7->notebook>=4.4.1-

>widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.2.3)

Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.

0->gmaps) (4.9.3)

Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/distpackages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidg ets>=7.0.0->gmaps) (4.11.2)

Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.

0->gmaps) (6.1.0)

Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/distpackages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidg ets>=7.0.0->gmaps) (0.7.1)

Requirement already satisfied: entrypoints>=0.2.2 in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid

getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.4)

Requirement already satisfied: jupyterlab-pygments in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid

getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.2.2)

Requirement already satisfied: MarkupSafe>=2.0 in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid

getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (2.1.3)

Requirement already satisfied: mistune<2,>=0.8.1 in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid

getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.8.4)

Requirement already satisfied: nbclient>=0.5.0 in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.8.0)

Requirement already satisfied: packaging in /usr/local/lib/python3.10/distpackages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidg ets>=7.0.0->gmaps) (23.2)

Requirement already satisfied: pandocfilters>=1.4.1 in

/usr/local/lib/python3.10/dist-packages (from nbconvert>=5->notebook>=4.4.1->wid getsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.5.0)

Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/distpackages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidg ets>=7.0.0->gmaps) (1.2.1)

Requirement already satisfied: fastjsonschema in /usr/local/lib/python3.10/distpackages (from nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps)

(2.18.1)

Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from

nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps)

(4.19.1)

Requirement already satisfied: argon2-cffi-bindings in

/usr/local/lib/python3.10/dist-packages (from argon2-cffi->notebook>=4.4.1->widg etsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (21.2.0)

Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/distpackages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->widgetsnbextension~=3

.6.0->ipywidgets>=7.0.0->gmaps) (23.1.0)

Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat->noteboo k>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (2023.7.1)

Requirement already satisfied: referencing>=0.28.4 in

/usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat->noteboo k>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (0.30.2)

Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/distpackages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->widgetsnbextension~=3

.6.0->ipywidgets>=7.0.0->gmaps) (0.10.6)

Requirement already satisfied: anyio<4,>=3.1.0 in

/usr/local/lib/python3.10/dist-packages (from jupyter-server>=1.8->nbclassic>=0.

4.7->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (3.7.1)

Requirement already satisfied: websocket-client in

/usr/local/lib/python3.10/dist-packages (from jupyter-server>=1.8->nbclassic>=0.

4.7->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.6.4)

Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.10/distpackages (from argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbexte nsion~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.16.0)

Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/distpackages (from beautifulsoup4->nbconvert>=5->notebook>=4.4.1->widgetsnbextension

~=3.6.0->ipywidgets>=7.0.0->gmaps) (2.5)

Requirement already satisfied: webencodings in /usr/local/lib/python3.10/distpackages (from bleach->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0-

>ipywidgets>=7.0.0->gmaps) (0.5.1)

Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.10/distpackages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclassic>=0.4.7->notebook> =4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (3.4)

Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/distpackages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclassic>=0.4.7->notebook>

=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.3.0)

Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/distpackages (from anyio<4,>=3.1.0->jupyter-server>=1.8->nbclassic>=0.4.7->notebook> =4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (1.1.3) Requirement already satisfied: pycparser in /usr/local/lib/python3.10/distpackages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1-> widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->gmaps) (2.21)

Building wheels for collected packages: gmaps

Building wheel for gmaps (setup.py) … done

Created wheel for gmaps: filename=gmaps-0.9.0-py2.py3-none-any.whl size=2076086

sha256=40b7725c2867fe885f2fe0fd33fff84621eb01e27185d11647a72f1cab3cc5a8

Stored in directory: /root/.cache/pip/wheels/b3/c2/dc/48b3ef16c2184dae51a003f1

7eb5d065bbbf1af3437d9f14e3

Successfully built gmaps

Installing collected packages: jedi, geojson, gmaps

Successfully installed geojson-3.0.1 gmaps-0.9.0 jedi-0.19.1

#In the Air Quality Analysis, incorporating Google Maps (GMap) can visually display pollution hotspots, helping users pinpoint areas with higher pollution levels. This feature enhances spatial understanding and facilitates informed decisions for better air quality management in Tamil Nadu.

[ ]: **import geopandas as gpd import matplotlib.pyplot as plt import pandas as pd**

*# Load geographic boundary data for Tamil Nadu (replace 'tamil\_nadu\_location.*

↪*shp' with the actual file path)* tamil\_nadu\_boundary = gpd.read\_file('/content/tamil\_nadu\_location.shp',␣ ↪encoding='utf-8')

*# Merge your data with the Tamil Nadu boundary data based on a common*␣

↪*identifier (e.g., location name)* merged\_data = tamil\_nadu\_boundary.merge(data, left\_on='NAME', right\_on='City/ ↪Town/Village/Area', how='right')

*# Create a map with the Tamil Nadu boundary data* ax = tamil\_nadu\_boundary.plot(figsize=(12, 8), color='lightgray',␣ ↪edgecolor='white')

*# Plot the locations and values on the map* merged\_data.plot(ax=ax, markersize=merged\_data['SO2'], alpha=0.1, legend=**True**,␣

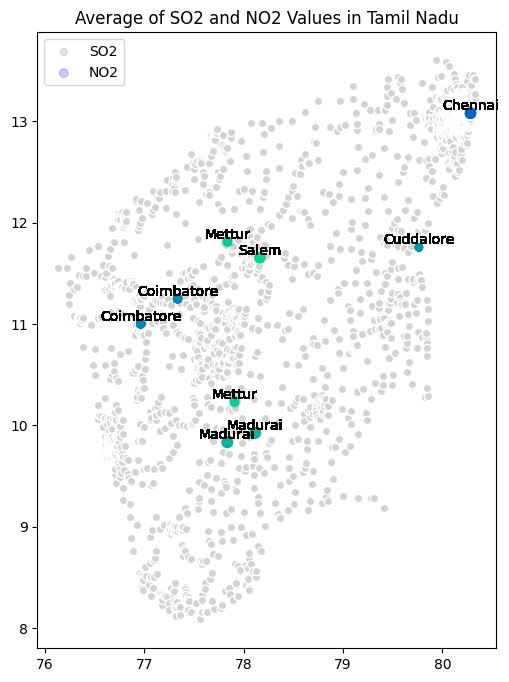
↪cmap='gist\_heat', label = "SO2") merged\_data.plot(ax=ax, markersize=merged\_data['NO2'], alpha=0.2, legend=**True**,␣ ↪cmap='winter', label = "NO2")

*# Add place names as labels to the points on the map* **for** x, y, label **in** zip(merged\_data.geometry.x, merged\_data.geometry.y,␣ ↪merged\_data['City/Town/Village/Area']):

**if not** pd.isna(x) **and not** pd.isna(y):

plt.annotate(label, (x, y), fontsize=10, ha='center', va='bottom')

plt.title('Average of SO2 and NO2 Values in Tamil Nadu') ax.legend() plt.show()



1. **Linear Regression Modeling:**
2. **We employed a simple linear regression model to estimate RSPM/PM10 levels based on SO2 and NO2 concentrations**
3. **Predict Respirable Suspended Particulate Matter (RSPM/PM10) levels**
4. **The segment demonstrates how to use a linear regression model to predict Respirable Suspended Particulate Matter (RSPM/PM10) levels based on Sulfur Dioxide (SO2) and Nitrogen Dioxide (NO2) levels. It loads and preprocesses the data, trains a linear regression model, and then predicts RSPM/PM10 levels for new data with specified SO2 and NO2 values.**

[27]: **import pandas as pd from sklearn.linear\_model import** LinearRegression

*# Load your air quality data into a Pandas DataFrame*

data = pd.read\_csv('/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv')

*# Preprocess the data and select relevant columns* data['Sampling Date'] = pd.to\_datetime(data['Sampling Date']) tn\_data = data[data['State'] == 'Tamil Nadu'] selected\_columns = ['SO2', 'NO2', 'RSPM/PM10']

tn\_data = tn\_data[selected\_columns].dropna() *# Remove rows with missing values*

*# Separate the features (SO2 and NO2) from the target (RSPM/PM10)*

X = tn\_data[['SO2', 'NO2']] y = tn\_data['RSPM/PM10']

*# Create and train a linear regression model*

model = LinearRegression() model.fit(X, y)

*# Now, you can use the trained model to make predictions for new data*

*# Replace 'new\_data' with the values of SO2 and NO2 you want to predict RSPM/*

↪*PM10 for* new\_data = [[100, 200]] *# Example values for SO2 and NO2* predicted\_rspm\_pm10 = model.predict(new\_data) print(f'Predicted RSPM/PM10: **{**predicted\_rspm\_pm10[0]**}** µg/m³')

Predicted RSPM/PM10: 331.29760210728915 µg/m³

<ipython-input-27-3d609d8cf440>:8: UserWarning: Parsing dates in DD/MM/YYYY format when dayfirst=False (the default) was specified. This may lead to inconsistently parsed dates! Specify a format to ensure consistent parsing.

data['Sampling Date'] = pd.to\_datetime(data['Sampling Date'])

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names

warnings.warn(

1. **Heatmap to visualize pollutant levels (RSPM/PM10)**
2. **To loads an air quality dataset and creates a heatmap to visualize pollutant levels (RSPM/PM10) by location and time in Tamil Nadu. It uses the Seaborn library for heatmap creation. The heatmap provides a graphical representation of air quality trends across different monitoring locations over time, aiding in identifying variations and hotspots in pollution levels.**

[ ]:

**import**

**pandas**

**as**

**pd**

**import**

**seaborn**

**as**

**sns**

**import**

**matplotlib**

**.**

**pyplot**

**as**

**plt**

*# Load your air quality dataset*

*# Replace 'your\_dataset.csv' with the actual file path*

df

=

pd

.

read\_csv(

'

/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv

'

)

*# Select the relevant columns for the heatmap (e.g., pollutant levels by*

␣

↪

*location and time)*

*# Replace 'Pollutant', 'Location', and 'Time' with your column names*

data

=

df

.

pivot\_table(index

=

'

Location of Monitoring Station

'

, values

=

'

RSPM/

↪

PM10

'

)

*# Create a heatmap*

plt

.

figure(figsize

=

(

12

,

8

))

*# Adjust the figure size as needed*

sns

.

heatmap(data, cmap

=

'

YlGnBu

'

, annot

=

**True**

, fmt

=

"

.1

f

"

)

*# Customize the heatmap labels and title*

plt

.

xlabel(

'

Time

'

)

plt

.

ylabel(

'

Location

'

)

plt

.

title(

'

Air Quality Heatmap

'

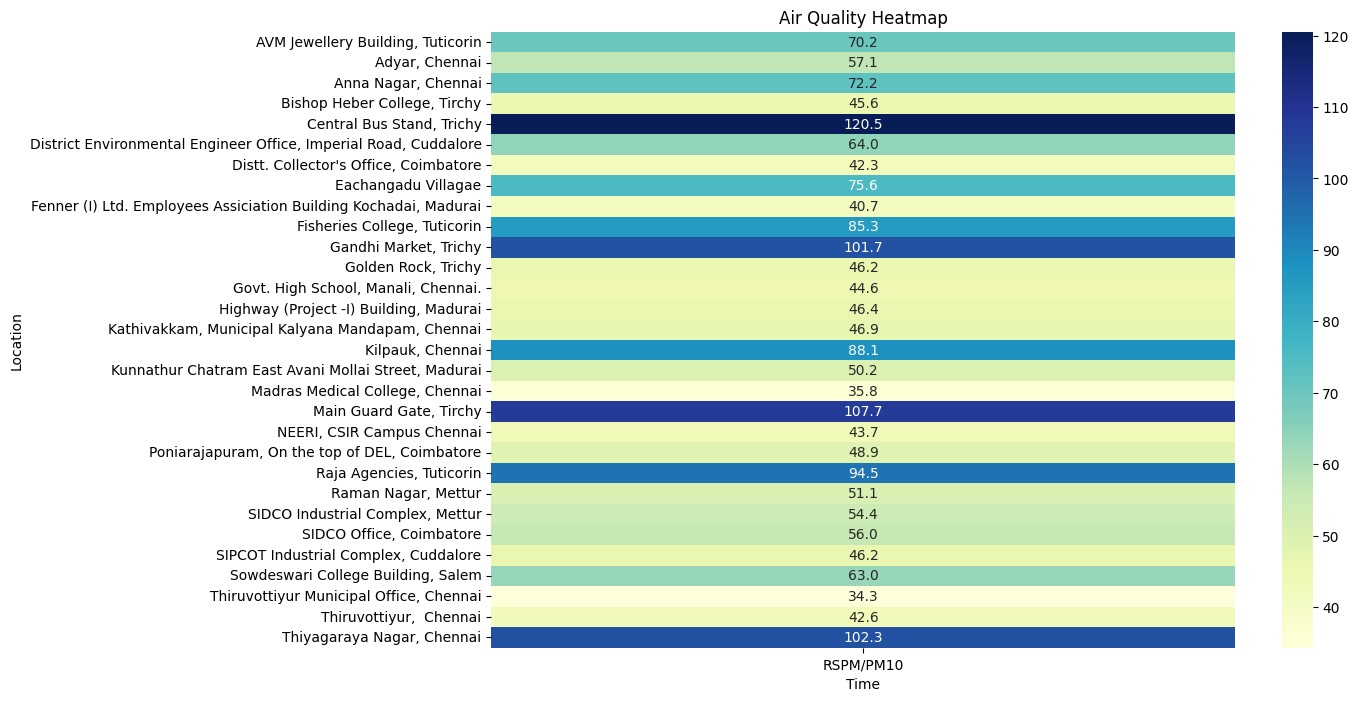
)

*# Display the heatmap*

plt

.

show()



[ ]:

**import**

**pandas**

**as**

**pd**

*# Load your CSV dataset into a DataFrame*

data

=

pd

.

read\_csv(

'

/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv

'

)

*# Group the data by the 'Region' column and calculate the mean for each group*

grouped

=

data

.

groupby(

'

City/Town/Village/Area

'

)[[

'

SO2

'

,

'

NO2

'

,

'

RSPM/PM10

'

]]

.

↪

mean()

*# Display the calculated averages*

print

(

grouped

)

SO2

NO2

RSPM/PM10

|  |  |  |  |
| --- | --- | --- | --- |
| City/Town/Village/Area |  |  |  |
| Chennai | 13.014042 | 22.088442 | 58.998000 |
| Coimbatore | 4.541096 | 25.325342 | 49.217241 |
| Cuddalore | 8.965986 | 19.710884 | 61.881757 |
| Madurai | 13.319728 | 25.768707 | 45.724490 |
| Mettur | 8.429268 | 23.185366 | 52.721951 |
| Salem | 8.114504 | 28.664122 | 62.954198 |
| Thoothukudi | 12.989691 | 18.512027 | 83.458904 |

Trichy

15.293956

18.695055

85.054496

[ ]:

**import**

**pandas**

**as**

**pd**

*# Load your CSV dataset into a DataFrame*

data

=

pd

.

read\_csv(

'

/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv

'

)

*# Group the data by the 'Region' column and calculate the mean for each group* grouped = data.groupby('Location of Monitoring Station')[['SO2', 'NO2', 'RSPM/ ↪PM10']].mean()

*# Display the calculated averages* print(grouped)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Location of Monitoring Station | SO2 NO2 | | | \ |
| AVM Jewellery Building, Tuticorin | 9.302083 12.697917 | | |  |
| Adyar, Chennai | 13.252174 18.965217 | | |  |
| Anna Nagar, Chennai | 13.873874 20.754545 | | |  |
| Bishop Heber College, Tirchy | 11.800000 14.942857 | | |  |
| Central Bus Stand, Trichy | 18.013333 21.506667 | | |  |
| District Environmental Engineer Office, Imperia… | 8.101010 19.151515 | | |  |
| Distt. Collector's Office, Coimbatore | 4.554348 25.793478 | | |  |
| Eachangadu Villagae | 11.916667 22.395833 | | |  |
| Fenner (I) Ltd. Employees Assiciation Building … | 13.643564 27.198020 | | |  |
| Fisheries College, Tuticorin | 14.526882 20.204301 | | |  |
| Gandhi Market, Trichy | 17.148649 20.797297 | | |  |
| Golden Rock, Trichy | 12.014085 15.000000 | | |  |
| Govt. High School, Manali, Chennai. | 13.043011 15.408602 | | |  |
| Highway (Project -I) Building, Madurai | 11.947917 24.458333 | | |  |
| Kathivakkam, Municipal Kalyana Mandapam, Chennai | 12.925532 15.170213 | | |  |
| Kilpauk, Chennai | 19.232759 27.172414 | | |  |
| Kunnathur Chatram East Avani Mollai Street, Mad… | 14.340206 25.577320 | | |  |
| Madras Medical College, Chennai | 7.418605 27.465116 | | |  |
| Main Guard Gate, Tirchy | 17.135135 20.837838 | | |  |
| NEERI, CSIR Campus Chennai | 5.931034 23.758621 | | |  |
| Poniarajapuram, On the top of DEL, Coimbatore | 4.126214 23.019417 | | |  |
| Raja Agencies, Tuticorin | 15.058824 22.441176 | | |  |
| Raman Nagar, Mettur | 7.572816 20.407767 | | |  |
| SIDCO Industrial Complex, Mettur | 9.294118 25.990196 | | |  |
| SIDCO Office, Coimbatore | 4.969072 27.329897 | | |  |
| SIPCOT Industrial Complex, Cuddalore | 6.969697 17.666667 | | |  |
| Sowdeswari College Building, Salem | 8.114504 28.664122 | | |  |
| Thiruvottiyur Municipal Office, Chennai | 8.360465 28.069767 | | |  |
| Thiruvottiyur, Chennai | 13.010417 15.583333 | | |  |
| Thiyagaraya Nagar, Chennai  Location of Monitoring Station | 18.849558 28.250000  RSPM/PM10 | | |  |
| AVM Jewellery Building, Tuticorin | 70.175258 | | |  |
| Adyar, Chennai | 57.068966 | | |  |
| Anna Nagar, Chennai | 72.187500 | | |  |
| Bishop Heber College, Tirchy | 45.633803 | | |  |
| Central Bus Stand, Trichy | 120.546667 | | |  |
| District Environmental Engineer Office, Imperia… | | 64.020202 |
| Distt. Collector's Office, Coimbatore | | 42.322222 |
| Eachangadu Villagae | | 75.591837 |
| Fenner (I) Ltd. Employees Assiciation Building … | | 40.732673 |
| Fisheries College, Tuticorin | | 85.255319 |
| Gandhi Market, Trichy | | 101.743243 |
| Golden Rock, Trichy | | 46.222222 |
| Govt. High School, Manali, Chennai. | | 44.612903 |
| Highway (Project -I) Building, Madurai | | 46.427083 |
| Kathivakkam, Municipal Kalyana Mandapam, Chennai | | 46.851064 |
| Kilpauk, Chennai | | 88.103448 |
| Kunnathur Chatram East Avani Mollai Street, Mad… | | 50.226804 |
| Madras Medical College, Chennai | | 35.837209 |
| Main Guard Gate, Tirchy | | 107.693333 |
| NEERI, CSIR Campus Chennai | | 43.678161 |
| Poniarajapuram, On the top of DEL, Coimbatore | | 48.883495 |
| Raja Agencies, Tuticorin | | 94.544554 |
| Raman Nagar, Mettur | | 51.106796 |
| SIDCO Industrial Complex, Mettur | | 54.352941 |
| SIDCO Office, Coimbatore | | 55.969072 |
| SIPCOT Industrial Complex, Cuddalore | | 46.171717 |
| Sowdeswari College Building, Salem | | 62.954198 |
| Thiruvottiyur Municipal Office, Chennai | | 34.310345 |
| Thiruvottiyur, Chennai | | 42.604167 |
| Thiyagaraya Nagar, Chennai | | 102.327434 |

[ ]: **import pandas as pd**

*# Load your CSV dataset into a DataFrame*

data = pd.read\_csv('/content/cpcb\_dly\_aq\_tamil\_nadu-2014 (1).csv')

*# Calculate the average pollution level for each area*

data['Average\_Pollution'] = data[['SO2', 'NO2', 'RSPM/PM10']].mean(axis=1)

*# Sort the areas in increasing order of average pollution levels* sorted\_data = data.sort\_values(by='Average\_Pollution',ascending = **False**)

*# Display the sorted DataFrame* print(sorted\_data[['City/Town/Village/Area', 'Average\_Pollution']])

City/Town/Village/Area Average\_Pollution

354 Chennai 113.500000

2636 Trichy 107.000000

438 Chennai 102.000000

2844 Trichy 100.333333

2846 Trichy 100.333333

… … …

1556 Cuddalore 11.333333

|  |  |  |
| --- | --- | --- |
| 849 | Chennai | 11.333333 |
| 1563 | Cuddalore | 11.333333 |
| 1557 | Cuddalore | 11.000000 |
| 1562 | Cuddalore | 10.666667 |

[2879 rows x 2 columns]