AIR QUALITY ANALYSIS AND PREDICTION

1. Introduction:

This document delves into a comprehensive analysis of air quality data in Tamil Nadu, aiming to provide valuable insights into pollutant levels and their trends. The document covers various aspects, including trend analysis, predictive modeling, and visualizations, offering a holistic view of air quality dynamics.

2. Average Pollution Levels Analysis:

Code:

```
# Calculate average pollution levels for each area

grouped = data.groupby('City/Town/Village/Area')[['SO2', 'NO2', 'RSPM/PM10']].mean()

print(grouped)
```

This section focuses on calculating average concentrations of SO2, NO2, and RSPM/PM10 across different monitoring stations, cities, or areas, providing a baseline understanding of typical pollution levels.

Output:

City/Town/Village/Area	SO2	NO2	RSPM/PM10
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

3. Trend Analysis:

3.1 Trends Over Time:

Code:

```
# Visualize trends in air pollution over time using seaborn

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

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')

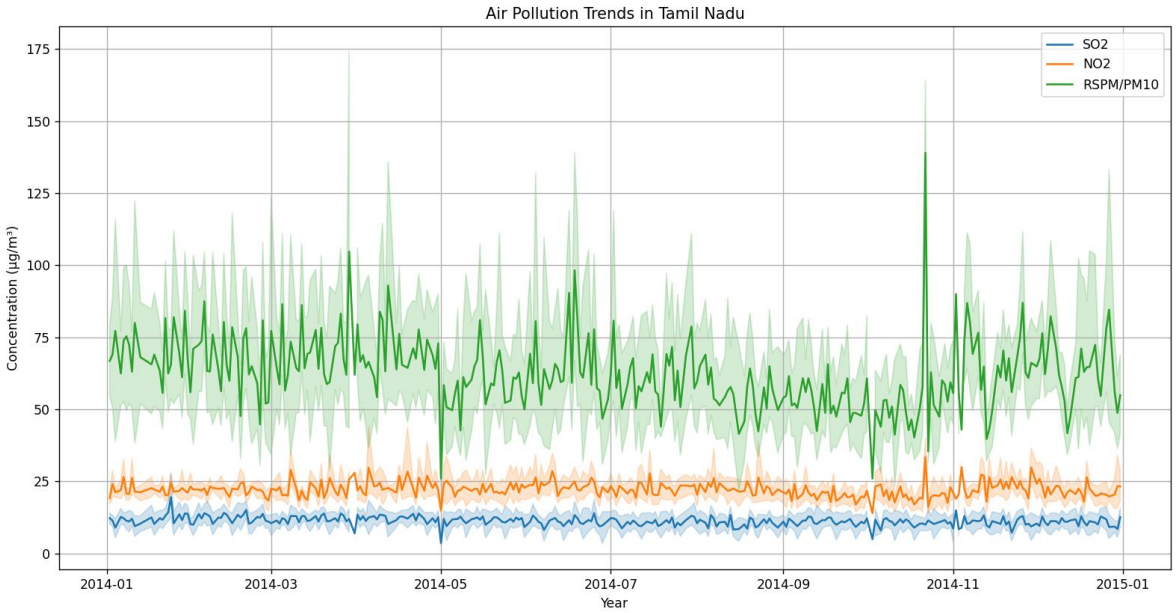
plt.legend()

plt.grid(True)

plt.show()

This section identifies trends in SO2, NO2, and RSPM/PM10 concentrations over the years, offering insights into the long-term evolution of air quality.

Output:



3.2 RSPM/PM10 Trends:

Code:

Plot trends in RSPM/PM10 pollution 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')

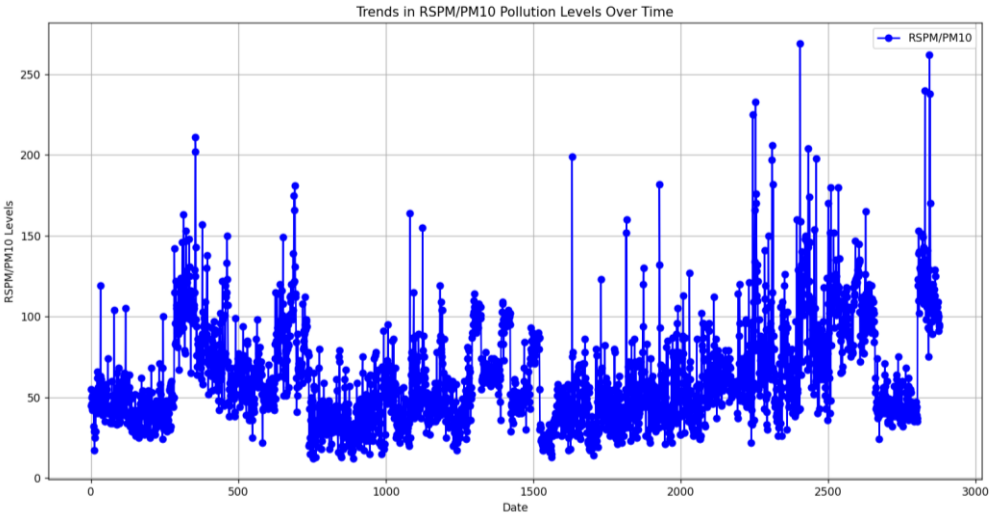
plt.grid(True)

plt.legend()

plt.show()

This specific plot zooms in on the trends in RSPM/PM10 pollution levels, offering a detailed perspective on their fluctuations.

Output:



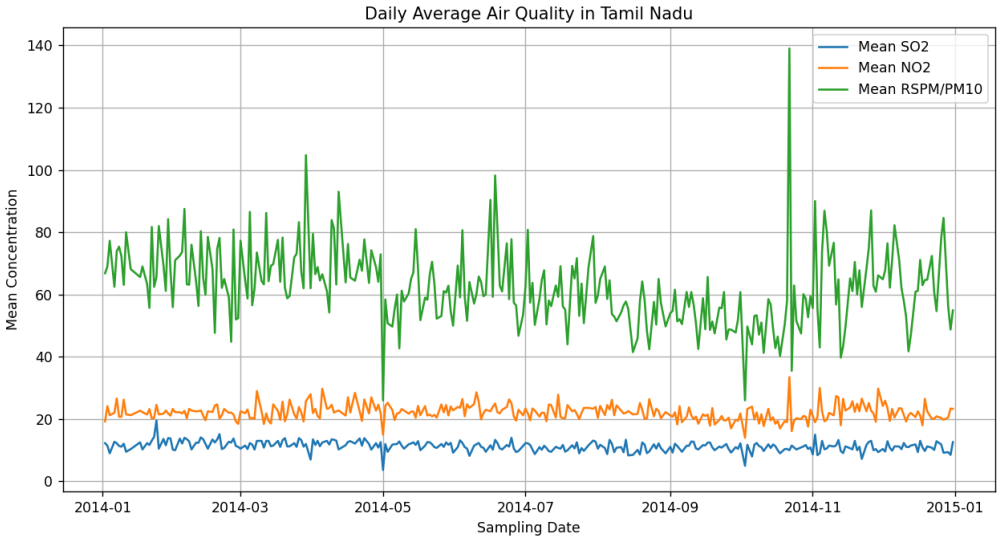
4. Daily Average Air Quality:

Code:

```
# Plot daily average air quality

daily_mean = data.groupby('Sampling Date').mean()

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()
```

This section visualizes the daily average concentrations of SO2, NO2, and RSPM/PM10, providing insights into day-to-day variations.

Output:



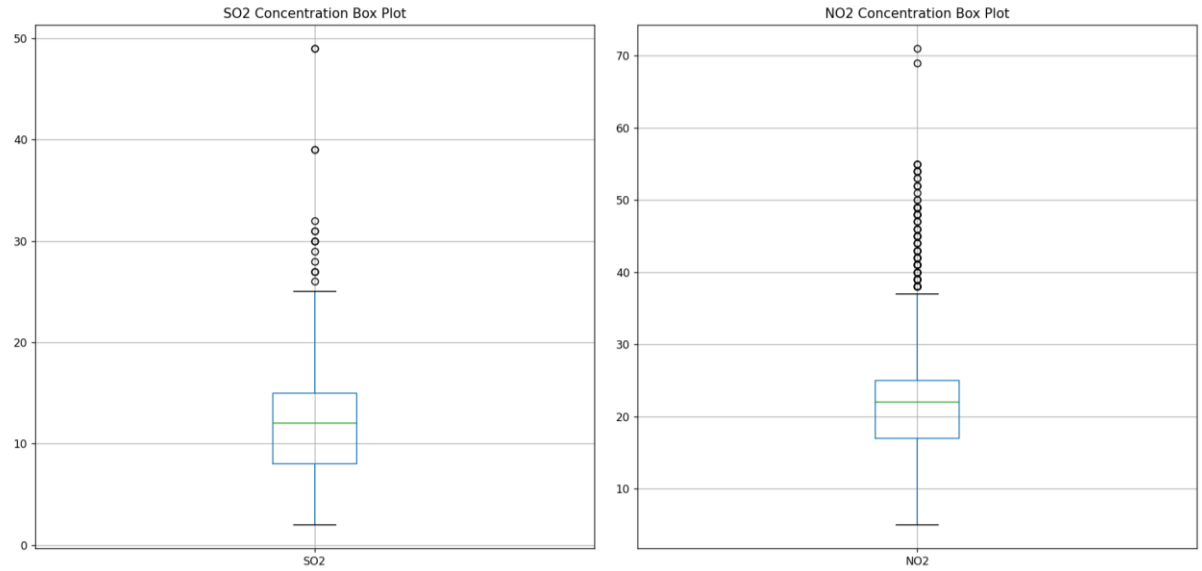
5. Box Plots and Descriptive Statistics:

Code:

```
# Display statistics and box plots for SO2 and NO2 concentrations

so2_stats = data['SO2'].describe()

no2_stats = data['NO2'].describe()

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()
```

This section includes descriptive statistics and box plots for SO2 and NO2 concentrations, offering a detailed view of the distribution and central tendency of these pollutants.

Output:



6.Predictive Modeling:

Code:

```
# Train a linear regression model to predict RSPM/PM10 levels
selected_columns = ['SO2', 'NO2', 'RSPM/PM10']
tn_data = tn_data[selected_columns].dropna()
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)
model = LinearRegression()
model.fit(X_train, y_train)
```

This section involves training a linear regression model to predict RSPM/PM10 levels based on SO2 and NO2 concentrations.

7 . User Input and Prediction:

Code:

```
# User Input for Prediction
user_so2 = float(input("Enter SO2 concentration (µg/m³): "))
user_no2 = float(input("Enter NO2 concentration (µg/m³): "))
```

```
# Predict RSPM/PM10 level based on user input
user_input = pd.DataFrame({'SO2': [user_so2], 'NO2': [user_no2]})
predicted_rspm_pm10 = model.predict(user_input[['SO2', 'NO2']])
print(f'Predicted RSPM/PM10 Level: {predicted_rspm_pm10[0]:.2f}')
```

This section allows users to input SO2 and NO2 concentrations, and the model predicts the corresponding RSPM/PM10 level.

Output:

```
Enter SO2 concentration (µg/m³): 10
Enter NO2 concentration (µg/m³): 20
Predicted RSPM/PM10 Level: 57.79
```

8. Model Evaluation:

Code:

```
# Model Evaluation
y_pred = model.predict(X_test)
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}')
```

This section evaluates the linear regression model's performance using Mean Squared

Output:

```
Mean Squared Error: 835.4788249190386
R-squared (Coefficient of Determination): 0.20658507746336507
```

9. Actual vs. Predicted Visualization:

Code:

```
# Visualization of Actual vs. Predicted
plt.figure(figsize=(8, 8))
plt.scatter(X_test['NO2'], y_test, color='blue', label='Actual', alpha=0.5)
plt.scatter(user_input['NO2'], predicted_rspm_pm10, color='red', marker='X', label='User Input')
```

plt.xlabel('NO2 Concentration')

plt.ylabel('RSPM/PM10 Levels')

plt.title('Actual vs. Predicted RSPM/PM10 Levels')

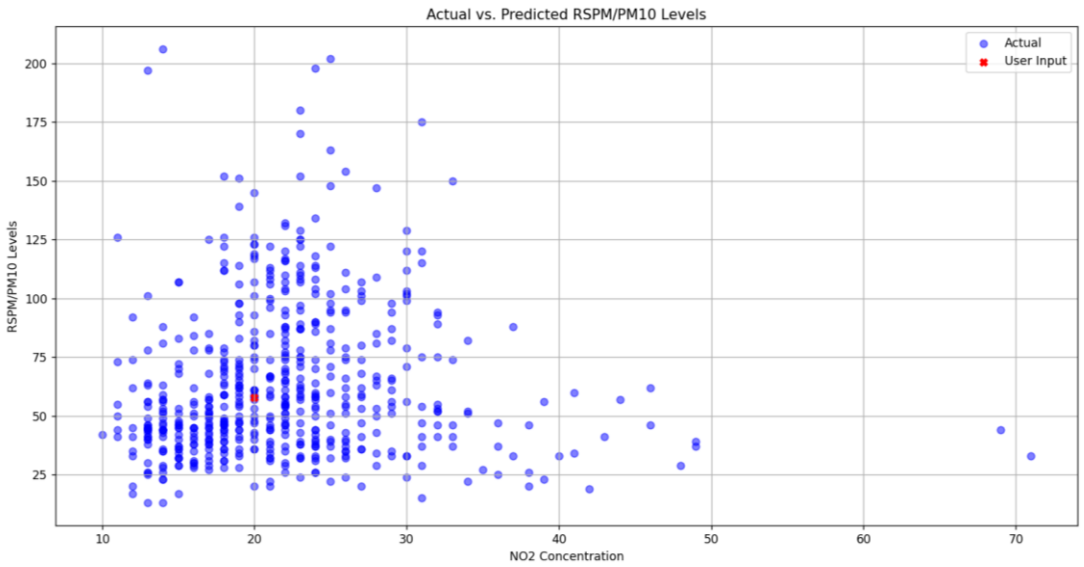
plt.legend()

plt.grid(True)

plt.show()

This plot compares actual RSPM/PM10 levels with predicted levels based on user input, providing an assessment of the model's performance.

Output:



10. Heatmap Visualization:

Code:

Visualize pollutant levels by location and time using a heatmap

df = pd.read_csv('cpcb_dly_aq_tamil_nadu-2014.csv')

data_heatmap = df.pivot_table(index='Location of Monitoring Station', values='RSPM/PM10')

plt.figure(figsize=(12, 8))

sns.heatmap(data_heatmap, cmap='YlGnBu', annot=True, fmt=".1f")

plt.xlabel('Time')

plt.ylabel('Location')

plt.title('Air Quality Heatmap')

plt.show()

This heatmap visualizes pollutant levels by location and time, providing a spatial perspective on air quality.

Output:

11. Conclusion:

In conclusion, this air quality analysis provides a multifaceted exploration of pollutant levels and trends in Tamil Nadu. The analysis begins with an assessment of average concentrations of SO2, NO2, and RSPM/PM10 across different regions, offering foundational insights into typical pollution levels. Trend analyses reveal the evolving landscape of air quality over time, with a focus on specific pollutants such as RSPM/PM10. The identified trends contribute to a deeper understanding of long-term patterns, potentially linking them to external factors. Daily average air quality visualizations showcase day-to-day variations in pollutant concentrations, offering valuable information for understanding short-term fluctuations. Additionally, the inclusion of box plots and descriptive statistics enhances our understanding of the distribution of SO2 and NO2 concentrations. The introduction of predictive modeling adds a predictive dimension to the analysis, allowing users to estimate RSPM/PM10 levels based on inputted SO2 and NO2 concentrations. The subsequent evaluation of the model's performance through mean squared error and R-squared values provides a quantitative measure of its accuracy. These findings collectively contribute to informed decision-making processes, facilitating the identification of high-pollution areas, understanding pollution trends, and ultimately guiding efforts to improve air quality in Tamil Nadu.