Predicting air quality levels using advanced Machine learning algorithms for environmental insights

Student Name:G.PRIYADHARSHINI

Register Number:412323243015

Institution: SRI RAMANUJAR ENGINEERING COLLEGE

Department: ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

Date of Submission: 10-05-2025

Github Repository Link: https://github.com/chandresh-29/Predicting-air-quality-levels-using-advanced-Machine-learning-algorithms-for-environmental-insights.git

---

1. Problem Statement

Refined Problem Definition

Air pollution is a pressing environmental and public health concern, affecting millions of people worldwide. Timely and accurate air quality predictions are essential for mitigating health risks and supporting policy decisions. Current air quality monitoring systems rely on physical sensors, which, while effective, can be expensive, sparse, and slow in delivering real-time insights. Advanced machine learning algorithms provide an opportunity to improve predictive accuracy, leveraging diverse datasets to generate reliable forecasts.

Understanding the Dataset

Building a robust prediction model requires analyzing features such as particulate matter (PM2.5, PM10), atmospheric conditions (temperature, humidity, wind speed), chemical pollutants (CO, SO_2, NO_2, O_3), and geographic factors. Refining the problem involves assessing the dataset's structure—handling missing values, removing irrelevant features, and selecting optimal variables for modeling.

Type of Problem

Predicting air quality falls under *regression analysis*, where the objective is to forecast air quality index (AQI) values based on historical and real-time data. Alternatively, classification models can be employed to categorize air quality levels into discrete risk levels (e.g., "Good," "Moderate," "Unhealthy").

Impact & Application Area

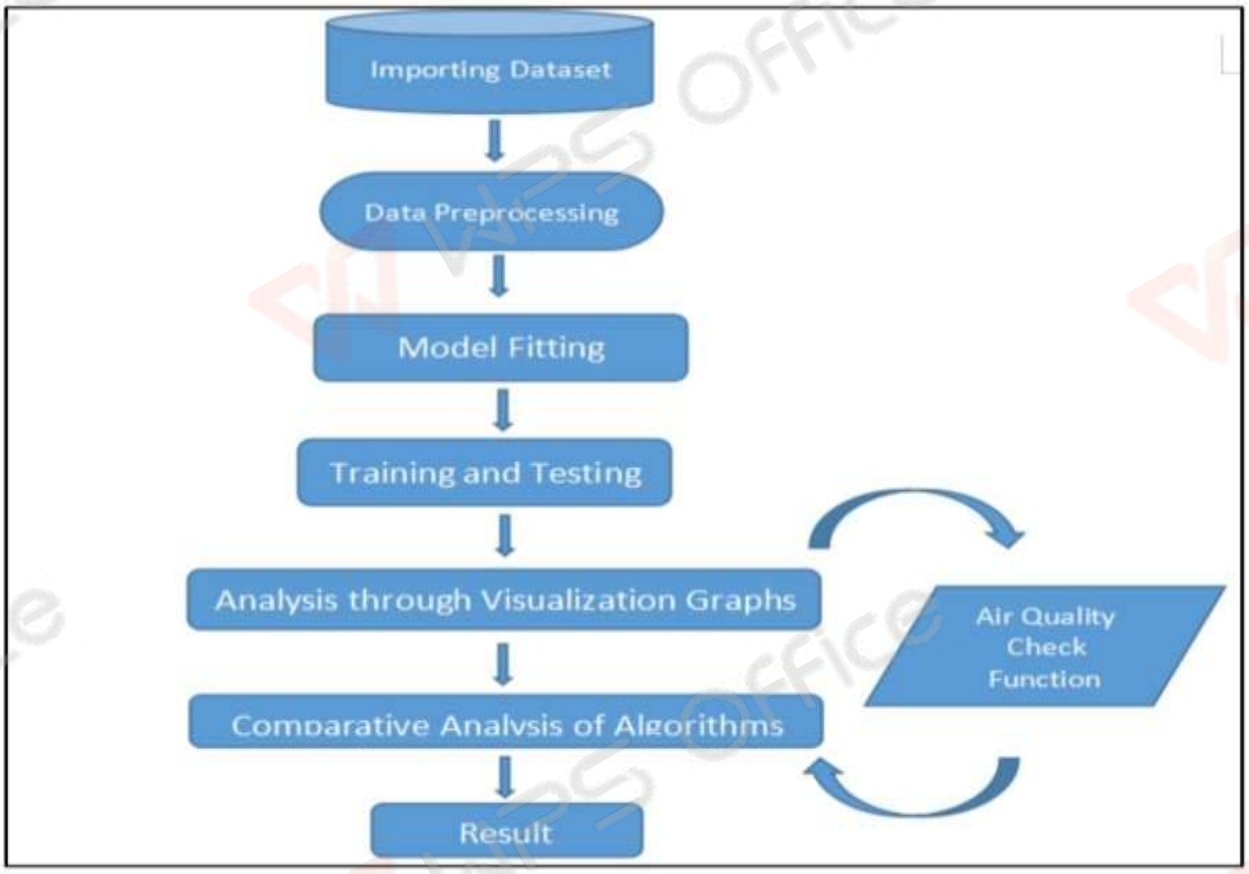
- **Public Health:** Reducing respiratory diseases and cardiovascular risks by informing individuals about air quality in advance.
- **Urban Planning & Policy:** Enabling governments to enforce pollution control measures and create sustainable environmental strategies.
- **Smart Cities & IoT:** Integrating machine learning-based air quality predictions into smart infrastructure, improving urban living conditions.
- **Transportation Management:** Adjusting traffic control strategies based on pollution forecasts to minimize congestion-related emissions.

2. Project Objectives

- To enhance prediction accuracy develop a machine learning model that delivers highly accurate AQI forecasts using diverse environmental and meteorological datasets.
- To improve interpretability ensure model transparency and explainability to help stakeholders understand the factors influencing air quality predictions.

- To real-world applicability design a scalable solution that can be integrated into smart cities, public health initiatives, and policy frameworks.
- To optimize feature selection refine input variables (pollutants, weather conditions) to boost model performance while minimizing unnecessary complexity.
- To deploy for accessibility implement the model through APIS (flask, fastapi) or cloud-based solutions for ease of access and real-time usage.
- To refine based on data exploration adjust modeling strategies post-data analysis, potentially incorporating deep learning architectures (lstm, xgboost, cnns) for improved predictions.

3. Flowchart of the Project Workflow



4. Data Description

- **Dataset Name & Origin**: Collected from sources like Github ,Kaggle, UCI Machine Learning Repository, or Open APIs (e.g., AQI datasets from government/environmental agencies).
- **Type of Data**: Structured, time-series data, including pollutant concentrations (PM2.5, PM10, CO, NO_2, SO_2, O_3), meteorological variables (temperature, humidity, wind speed), and location-based attributes.
- **Number of Records & Features**: Typically consists of hundreds of thousands of records with 10-20 key features influencing air quality predictions.
- **Static or Dynamic Dataset**: Primarily dynamic, as air quality fluctuates over time, requiring continuous updates from real-time sensors or APIs.
- **Target Variable**: AQI (Air Quality Index) for supervised learning, predicting pollution levels based on historical and real-time data.

5. Data Preprocessing

1. Handling Missing Values

- Identify missing values in AQI, pollutant, and meteorological data.
- Impute missing values using mean, median, mode, or forward-fill for timeseries continuity.
- Drop rows with excessive missing data if they reduce prediction reliability.

2. Removing Duplicate Records

- Check for duplicates based on timestamp, location, and pollutant levels.
- Keep only unique entries to prevent redundancy and bias in modeling.

3. Detecting & Treating Outliers

- Use Z-score analysis or IQR method to detect extreme values in pollutants.
- Apply log transformation or winsorization to handle skewed data.

4. Data Type Conversion & Consistency

- Convert timestamps to datetime format for efficient time-series processing.
- Ensure all numerical features (AQI, pollutant concentrations) are properly formatted as floats.

5. Encoding Categorical Variables

- Label Encoding for ordinal air quality categories (e.g., Good, Moderate, Unhealthy).
- One-Hot Encoding for categorical location-based features to prevent model bias.

6. Feature Scaling & Normalization

- Use Min-Max scaling or Standardization (Z-score normalization) for pollutant concentrations and meteorological variables to ensure consistent input ranges for ML models.

7. Documentation

- Maintain structured markdown explanations in the code to document each preprocessing step.
- Ensure transformation steps are reproducible and explainable for future model iterations.

6. Exploratory Data Analysis (EDA)

1. Univariate Analysis

- **Histograms**: Assess pollutant distribution (PM2.5, NO_2, CO) to identify skewness and peaks.
- **Boxplots**: Detect anomalies and outliers in pollutant concentration levels.
- **Countplots:** Visualize categorical distributions, such as air quality levels.

2. Bivariate & Multivariate Analysis

- **Correlation Matrix**: Determine relationships between pollutants and meteorological variables.
- **Pairplots:** Explore patterns in pollutant interactions.
- **Scatterplots**: Analyze AQI trends with temperature, humidity, and wind speed.
- **Grouped Bar Plots**: Compare air quality changes across locations and time frames.

3. Analysis of Feature-Target Relationships

- Evaluate AQI dependence on temperature, humidity, and wind conditions.
- Analyze pollutant interactions (e.g., higher NO_2 levels correlating with CO spikes).
- Identify leading contributors to AQI fluctuations for predictive modeling.

4. Insights Summary

- Strong correlations between particulate matter (PM2.5, PM10) and AQI fluctuations.
- Seasonal variations affecting pollution trends (e.g., winter inversion trapping pollutants).

- **Geographical influence**: Urban areas exhibit higher pollution due to vehicular emissions.

7. Feature Engineering

1. Creating New Features Based on Domain Knowledge

- **Air Quality Trends**: Compute rolling averages (e.g., 7-day AQI moving average) to capture temporal trends.
- **Weather-Pollution Interaction**: Introduce features combining meteorological conditions, such as Temperature-Humidity Index, influencing pollution dispersion.
- **Traffic Influence**: If available, integrate traffic congestion metrics as a proxy for vehicle emissions.

2. Transforming Existing Features

- **Datetime Features**: Extract hour, day, month, season from timestamps to capture cyclical pollution variations.
- **Pollutant Ratios:** Compute ratios like NO_2/CO to identify combustion related pollution sources.
- **Binning AQI Values**: Categorize AQI into discrete health risk levels for classification-based predictions.

3. Applying Dimensionality Reduction (Optional)

- **PCA (Principal Component Analysis):** Reduce redundancy in correlated features (e.g., PM2.5 & PM10).

- **t-SNE for Visualization:** Explore high-dimensional pollutant interactions in a 2D space.

4. Justifying Feature Engineering Choices

- Each transformation aligns with real-world pollution behavior, enhancing predictive accuracy.
- New features improve interpretability, helping stakeholders derive actionable insights.
- Dimensionality reduction aids in handling multicollinearity and optimizing model performance.

8. Model Building

1. Selected Models & Justification

- **XGBoost:** A robust boosting algorithm known for its high accuracy and ability to handle missing values. Suitable for air quality regression due to its capability in capturing nonlinear relationships.
- **LSTM (Long Short-Term Memory):** Effective for time-series forecasting, capturing temporal dependencies in AQI fluctuations. Ideal for dynamic environmental data.

2. Data Splitting Strategy

- **Train-Test Split:** 80-20 ratio to ensure balanced model evaluation.
- **Time-Based Stratification:** Preserves chronological order to align with realworld forecasting needs.

3. Training & Initial Evaluation

XGBoost Regression Metrics:

- **MAE (Mean Absolute Error):** Measures absolute differences in AQI predictions.
- **RMSE (Root Mean Square Error:** Penalizes large errors, ensuring stability in high pollution events.
- **R^2 Score:** Assesses variance explanation by the model.

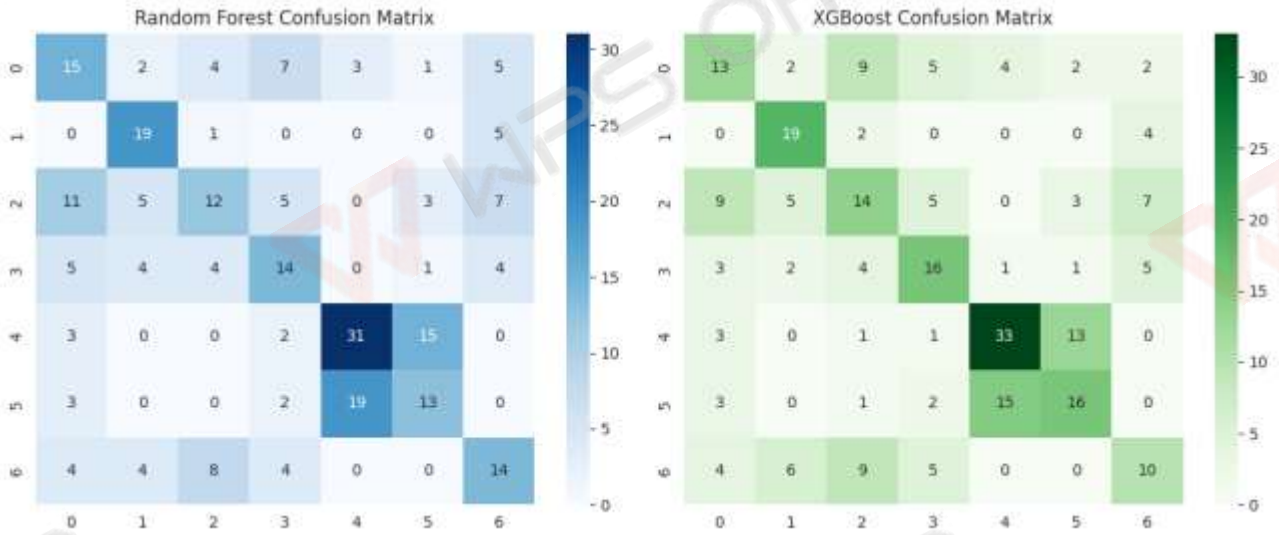
LSTM Regression Metrics:

- Same as above, but with additional tuning for sequence learning (e.g., adjusting timestep size).

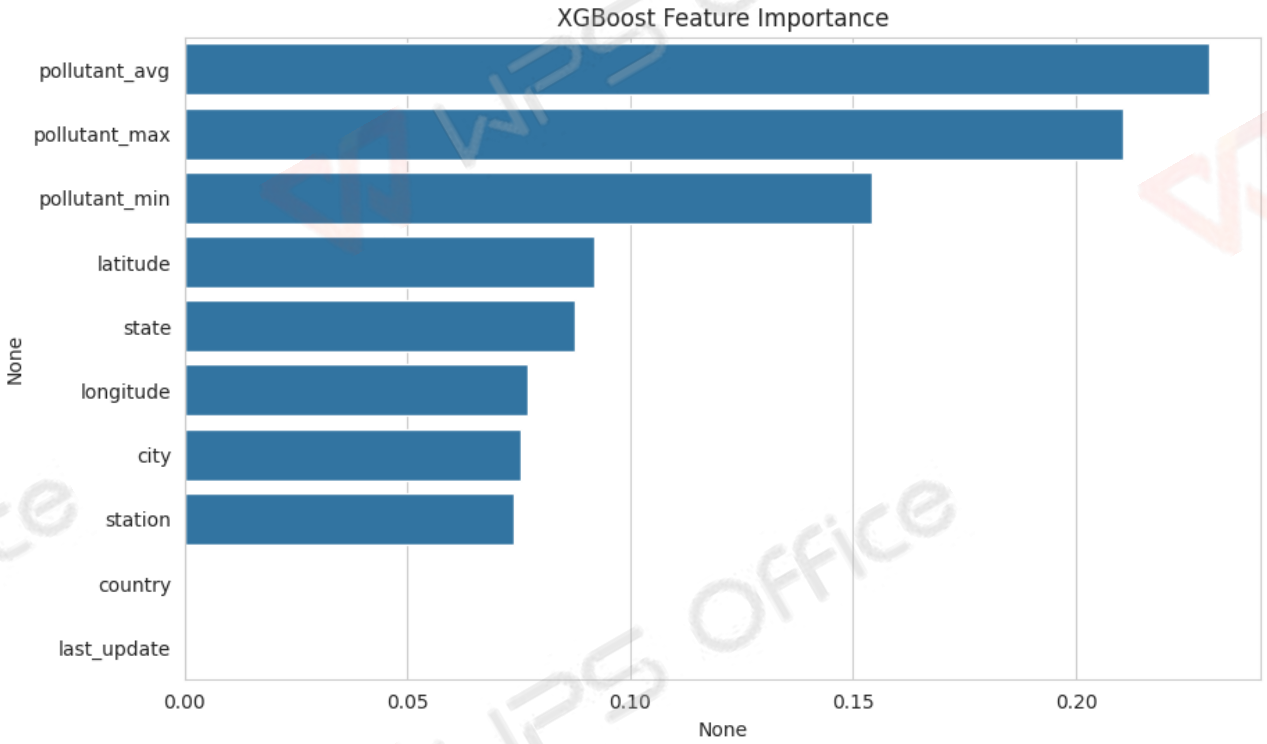
4. Model Comparison & Optimization

- Compare XGBoost's feature importance approach with LSTM's long-term dependency modeling.
- Hyperparameter tuning using Grid Search (XGBoost) and learning rate adjustments (LSTM.
- Experiment with Ensemble Models (e.g., combining XGBoost & LSTM for hybrid predictions).

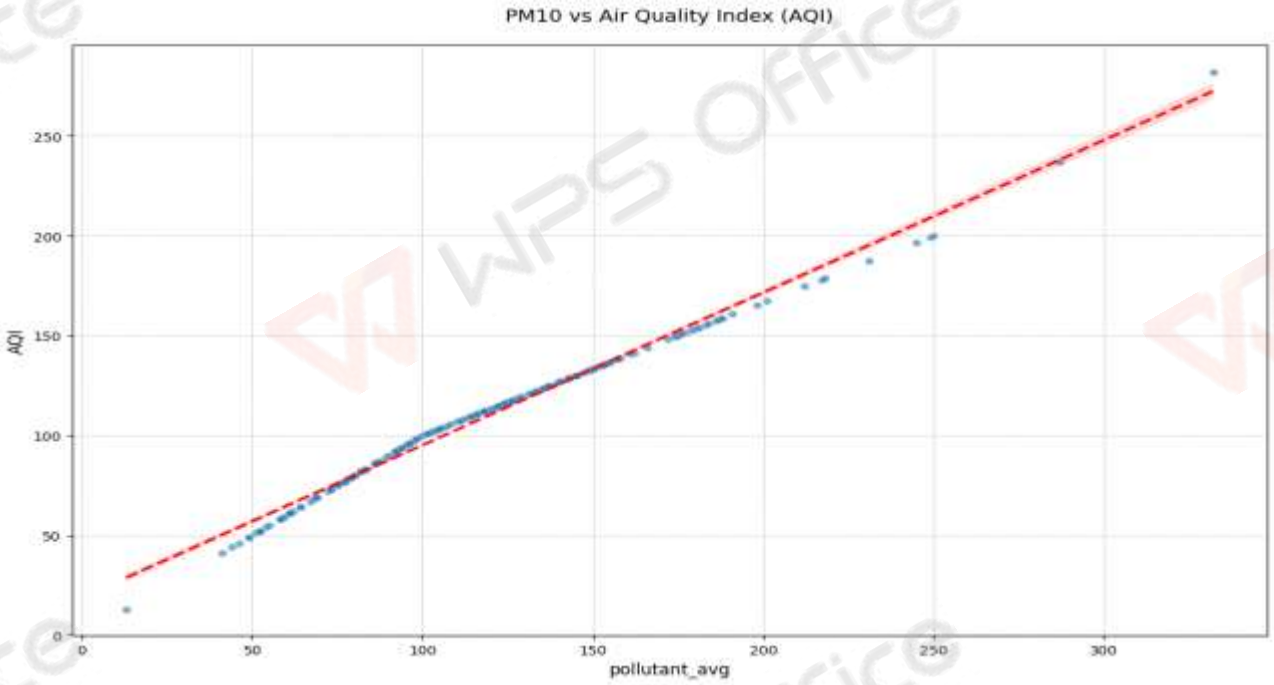
9. Visualization of Results & Model Insights



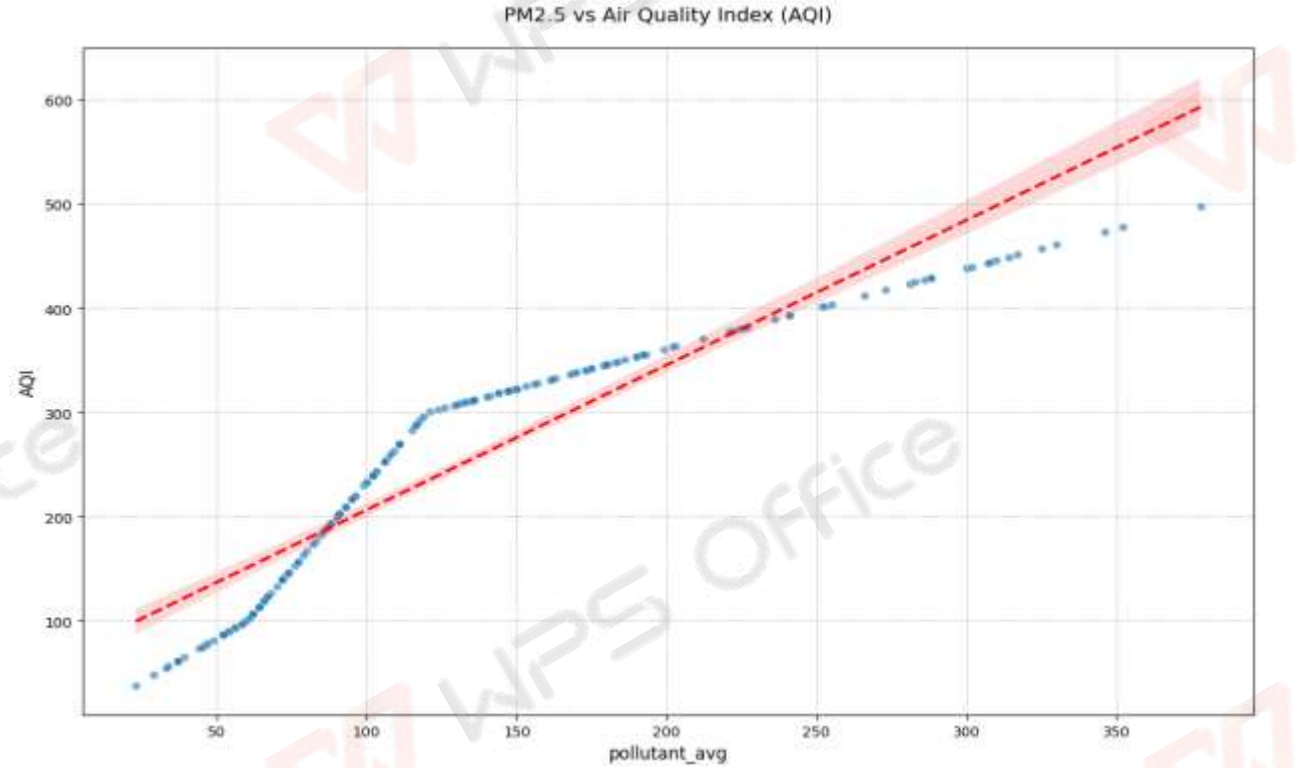
Visualization of random and XGBoost confusion matrix



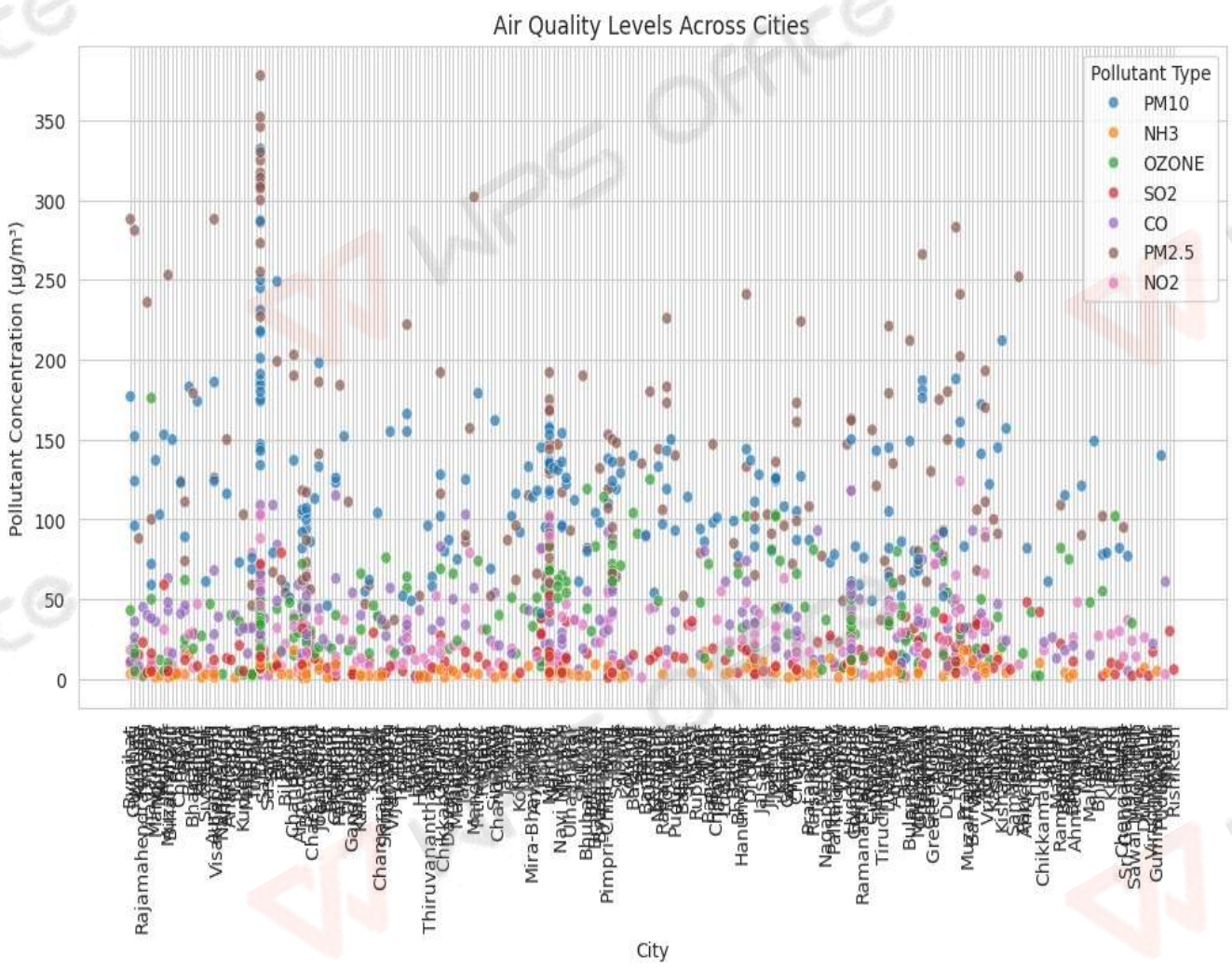
Visualization of XGBoost Feature Importance



Graph for PM10 vs Air Quality Index



Graph for PM2.5 vs Air Quality Index



Graph for Air Quality Levels Across Cities

10. Tools and Technologies Used

Programming Languages:

- Python (preferred for ML, data science, and ecosystem support)
- R (strong for statistical analysis)

Development Environments:

- Google Colab (cloud-based, quick experimentation)
- Jupyter Notebook (interactive data exploration)
- VS Code (lightweight and extensible)

Core Libraries for Data Processing & Modeling:

- pandas (data manipulation)
- numpy (numerical computations)
- seaborn, matplotlib, plotly (visualization)
- scikit-learn (machine learning models)
- XGBoost (powerful gradient boosting for tabular data)

Visualization & Dashboarding Tools:

- Plotly (interactive graphs)
- Tableau (for detailed environmental reports)
- Power BI (data-driven insights)

11. Team Members and Contributions

1. **CHANDRESH P –** [FEATURE ENGINEERING \ EDA]
   Role: To Transforming Raw Data into Meaningful Inputs and To Understanding Air Quality Data Before Modelling.

2. **SANJAI KUMARAN M –** [DOCUMENTATION \ REPORTING]
   Role: To Record Details of Machine Learning Algorithms and Compiling and Presenting Environmental Insights Derived from Model Outputs.

3. **PRIYADHARSINI G –** [DATA CLEANING \ MODEL DEVELOPMENT]
   Role: To Ensuring the Quality and Accuracy of Environmental Data and Designing, Testing, and Advanced Machine Learning Algorithms to Generate Reliable Predictions and Insights.