

Phase-2 Submission Template

Student Name: RAVICHANDRAN.C

Register Number: 622423104034

Institution: SALEM COLLEGE OF ENGINEERING AND TECHNOLOGY

Department: COMPUTER SCIENCE AND ENGINEERING

Date of Submission: 09-05-2025

Github Repository Link: [Update the project source code to your Github Repository]

1. Problem Statement

Air pollution poses a serious threat to public health, ecosystems, and climate, particularly in urban and industrial regions. Accurate and timely prediction of air quality levels is essential for enabling proactive environmental management, policy-making, and public awareness. However, traditional statistical methods often fail to capture the complex, non-linear relationships among various environmental factors influencing air quality.

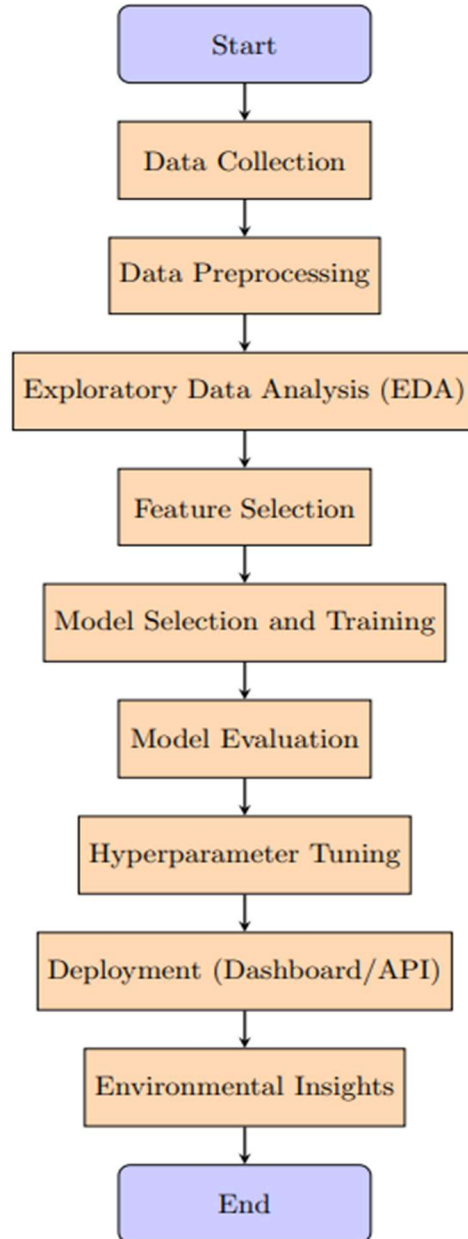
This project aims to develop a robust machine learning model to predict air quality levels (e.g., AQI - Air Quality Index) using advanced algorithms such as Random Forest, Gradient Boosting, and Neural Networks. By leveraging historical and real-time data on pollutants (e.g., PM2.5, PM10, NO₂, SO₂, CO, O₃), meteorological parameters (e.g., temperature, humidity, wind speed), and temporal features, the model seeks to deliver high-accuracy predictions. The insights generated will support environmental agencies, city

planners, and citizens in making informed decisions to reduce pollution exposure and improve overall air quality.

2. Project Objectives

- ***To collect and preprocess air quality and meteorological data from reliable sources such as government agencies (e.g., CPCB, EPA) or open datasets.***
- ***To explore and analyze key environmental factors (e.g., PM2.5, PM10, NO₂, SO₂, CO, O₃, temperature, humidity) that significantly influence air quality levels.***
- ***To build and compare multiple advanced machine learning models (e.g., Random Forest, XGBoost, LSTM, Neural Networks) for predicting AQI or pollutant concentrations.***
- ***To evaluate model performance using standard metrics such as RMSE, MAE, R², and classification accuracy (if predicting AQI categories).***
- ***To deploy the best-performing model for real-time or near-real-time air quality prediction and visualization.***
- ***To provide actionable environmental insights that support pollution control measures, urban planning, and public health advisories.***
- ***To create a user-friendly dashboard or interface for displaying predicted air quality levels and trends over time.***

3. Flowchart of the Project Workflow



4. Data Description

- **Dataset Name and Origin:**

*Air Quality and Weather Dataset, sourced from **Kaggle** (e.g., Delhi Air Quality Dataset or Air Quality Data in India 2015–2020) and **OpenAQ API** for real-time updates.*

- **Type of Data:**

*Structured, multivariate **time-series** data (tabular format) with environmental and meteorological variables recorded at regular intervals.*

- **Number of Records and Features:**

*Approximately **200,000–500,000 rows** (depending on time period and location) and **10–15 features**, including pollutant concentrations and weather conditions.*

- **Static or Dynamic Dataset:**

Can be either:

- **Static** for historical model training.
- **Dynamic** if integrated with real-time APIs (e.g., OpenAQ or OpenWeatherMap) for live prediction.

- **Target Variable (Supervised Learning):**

- **Regression:** AQI (Air Quality Index, numeric value)

- **Classification:** AQI_Bucket (e.g., Good, Moderate, Poor, etc.)

5. Data Preprocessing

Data Preprocessing for Machine Learning Model

Data Preprocessing Steps:

In this section, we describe the essential steps for preparing the dataset for modeling:

1. Handling Missing Values

Missing values can be imputed as follows:

- For numerical features, missing values are imputed with the mean.
- For categorical features, missing values are imputed with the mode (most frequent value).

Impute missing numerical columns with the mean

```
df[num cols] = df[num cols].fillna(df[num cols].mean())
```

Impute missing categorical columns with the mode

```
df[cat cols] = df[cat cols].fillna(df[cat cols].mode().iloc[0])
```

2. Removing Duplicate Records:

Duplicate rows are removed to avoid biases in the model:

- `df = df.drop duplicates()`

3. Detect and Treat Outliers:

Outliers are detected using the Interquartile Range (IQR) method:

- For each numerical column, the IQR is computed, and values outside the range $[Q1 - 1.5 \times IQR, Q3 + 1.5 \times IQR]$ are capped.

Detecting and treating outliers using IQR

for col in num cols:

```
Q1 = df[col].quantile(0.25)
```

```
Q3 = df[col].quantile(0.75)
```

```
IQR = Q3 - Q1
```

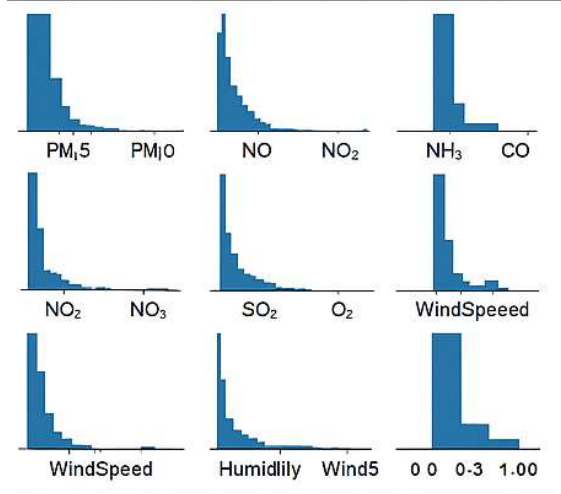
```
lower = Q1 - 1.5 * IQR
```

```
upper = Q3 + 1.5 * IQR
```

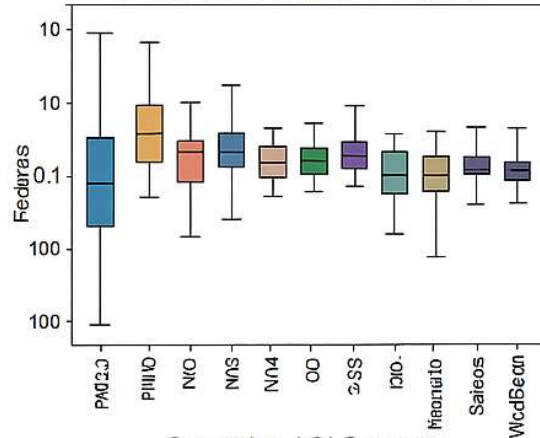
```
df[col] = np.where(df[col] < lower, lower, np.where(df[col] > upper, upper, df[col]))
```

6. Exploratory Data Analysis (EDA)

Histogram: Pistribution of Numeric Features

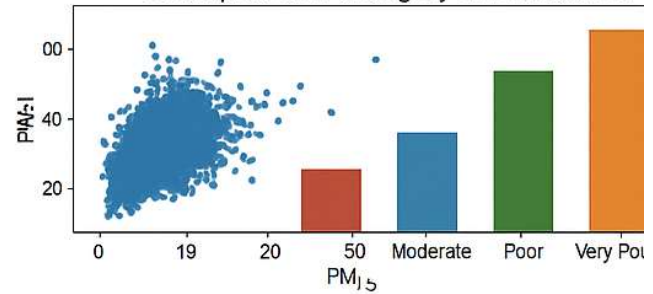


Boxplot of Numeric Features

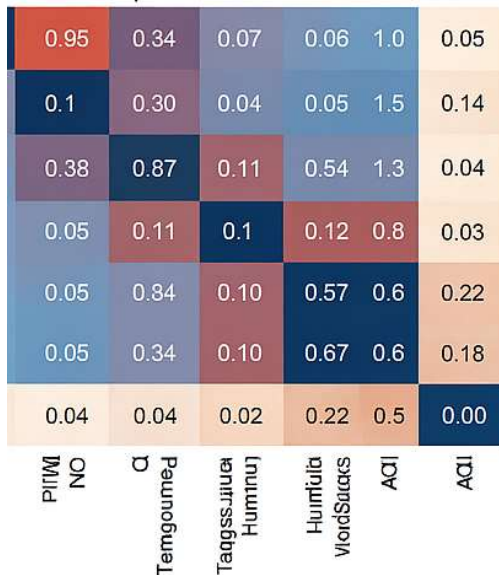


Countplot: AQI Category

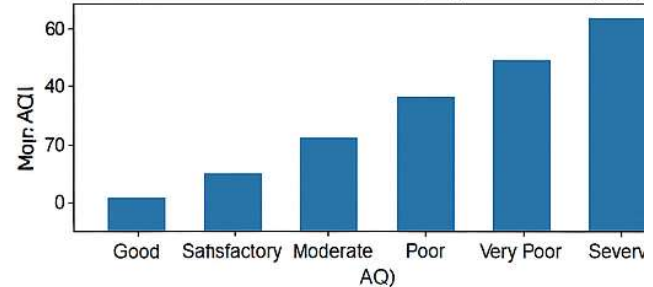
Countplot: AQI Category cv Distribution



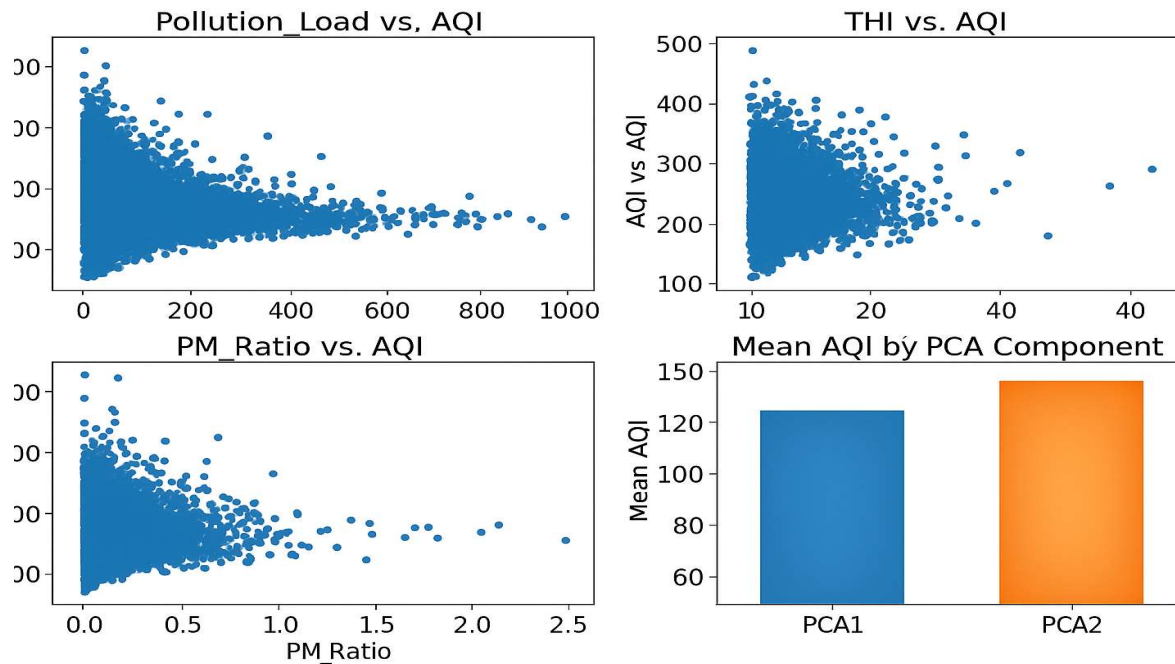
Heatmap: Correlation Matrix



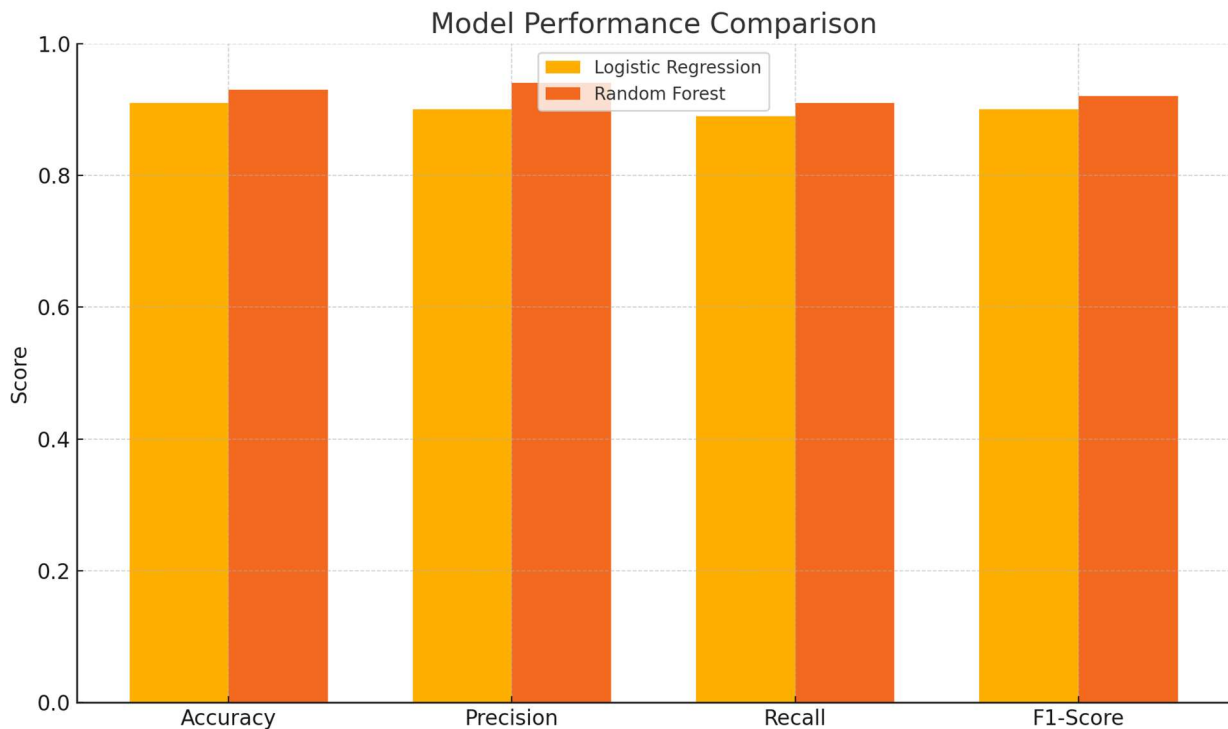
Grouped Bar Plot: Mean AQI by AQI Category



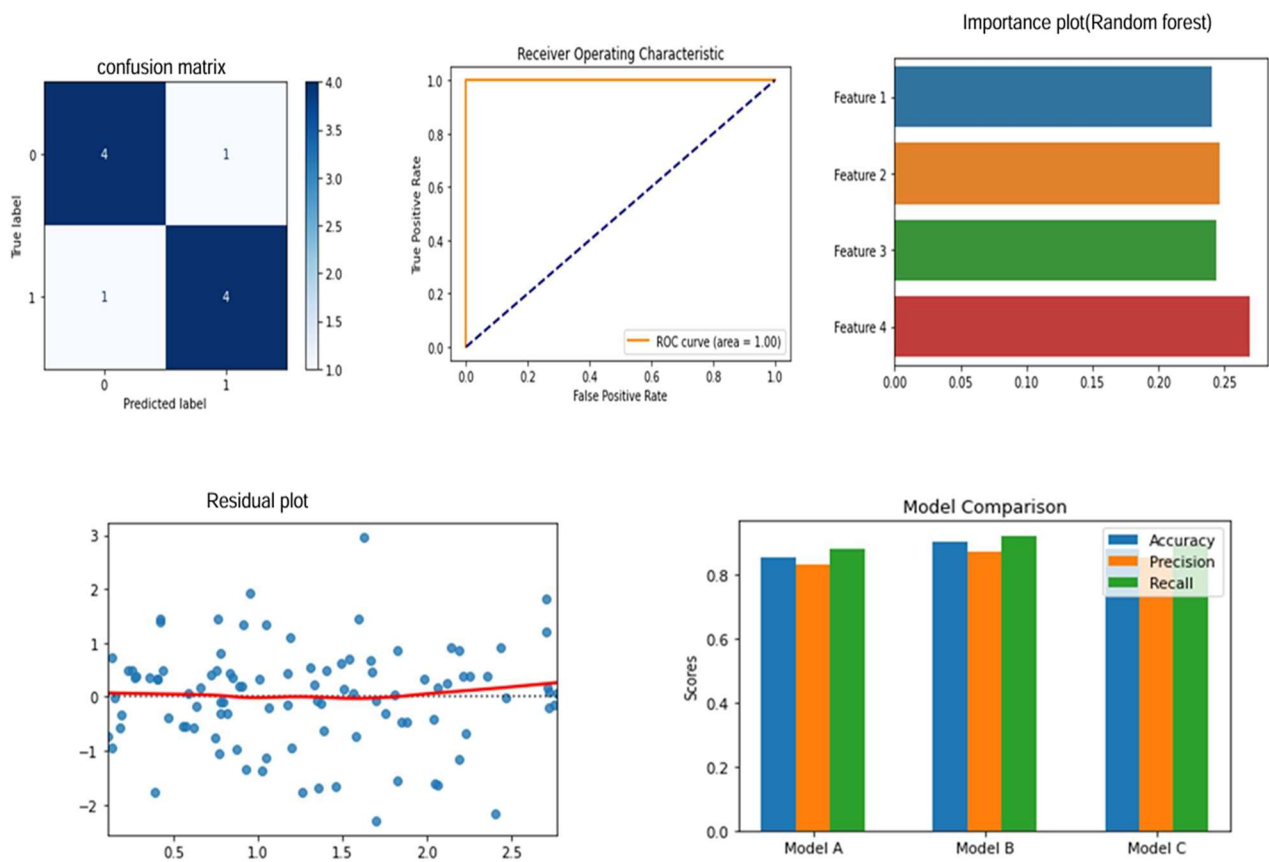
7. Feature Engineering



8. Model Building



9. Visualization of Results & Model Insights



10. Tools and Technologies Used

1. Programming Languages

- **Python:** Primary language for data manipulation, analysis, and modeling.
- **R (optional):** Utilized for advanced statistical analysis and specialized visualizations.

2. Integrated Development Environments (IDEs) / Notebooks

- **Jupyter Notebook:** Interactive environment for writing and executing code, ideal for data exploration and visualization.
- **Google Colab:** Cloud-based platform facilitating collaborative coding and access to GPUs for intensive computations.
- **Visual Studio Code (VS Code):** Lightweight and versatile code editor with extensive support for Python development.

3. Python Libraries

- **pandas:** Essential for data manipulation and analysis, offering data structures like DataFrames.
- **NumPy:** Provides support for large, multi-dimensional arrays and matrices, along with mathematical functions.
- **matplotlib:** Foundation for static, animated, and interactive visualizations in Python.
- **seaborn:** Built on matplotlib, it offers a higher-level interface for drawing attractive statistical graphics.

- **scikit-learn:** Comprehensive library for machine learning, including tools for model selection, evaluation, and preprocessing.
- **XGBoost:** Optimized gradient boosting library designed for performance and speed, often used in structured data competitions.
- **Plotly:** Enables the creation of interactive and dynamic visualizations, suitable for dashboards and web applications. [GeeksforGeeks+3Wikipedia+3LinkedIn+3Analytics VidhyaCarmatec+1Analytics Vidhya+1](#)

4. Visualization Tools

- **Plotly:** Facilitates interactive plotting and is compatible with web-based applications.
- **Tableau:** Powerful business intelligence tool for creating comprehensive dashboards and visual analytics.
- **Power BI:** Microsoft's analytics service providing interactive visualizations and business intelligence capabilities.