**Phase-2**

**Student Name:Subhashini K**

**Register Number:** 613023104107

**Institution:** vivekanandha college of technology for women

**Department:** B.E.Computer science and engineering

**Date of Submission:** 01.05.2025

**Github Repository Link:** [**https://github.com/Subhashini-200-600/nm-phase2**](https://github.com/Subhashini-200-600/nm-phase2)

### **1. Problem Statement**

* Air pollution poses significant threats to public health and the environment, yet accurate real-time prediction of air quality levels remains a challenge due to the complexity and variability of contributing factor.
* This project aims to develop a robust machine learning model that can predict air quality levels based on environmental, meteorological, and pollution data.
* By leveraging advanced algorithms, the system seeks to provide timely and accurate air quality insights to support environmental policy, public health decisions, and individual awareness

### 

### **2. Project Objectives**

* collection and preprocessing:Gather and clean historical air quality, meteorological, and pollutant data from reliable sources to create a comprehensive dataset for model training.
* Feature Engineering and Selection:Identify and construct relevant features that influence Data air quality, such as temperature, humidity, wind speed, industrial emissions, and traffic levels.
* Model Development:Develop and compare multiple advanced machine learning models (e.g., Random Forest, XGBoost, LSTM, or CNN) to predict air quality index (AQI) levels.

### **3. Flowchart of the Project Workflow**

### C:\Users\dhana\AppData\Local\Packages\5319275A.WhatsAppDesktop_cv1g1gvanyjgm\TempState\0C13D6D3B3743756E05BD42F3AFB2D52\WhatsApp Image 2025-05-01 at 11.55.21_f00e88f9.jpg

### 

### 

### **4. Data Description**

### The dataset used in this project contains air quality measurements collected from various monitoring stations across different geographic locations. The data includes both meteorological parameters and concentrations of key air pollutants. The primary goal is to predict the Air Quality Index (AQI) or specific pollutant levels based on environmental and temporal features*.*

### **5. Data Preprocessing**

* Combine datasets from multiple stations or sources.Ensure uniform time formats and units across data entries.
* Remove rows with extensive missing data.
* Extract hour, day, month, weekday, or season from timestamps.

Create lag features to capture temporal patterns (e.g., previous day’s PM2.5).

* Standardize features (e.g., temperature, pressure, pollutant levels) using:Min-Max ScalingZ-score Standardization.Especially important for distance-based or gradient-based algorithms (e.g., SVM, neural networks).
* If predicting AQI category, encode target labels (e.g., Good, Moderate, Unhealthy).
* If predicting AQI or pollutant levels (regression), ensure numerical continunity
* Use SMOTE, oversampling, or undersampling if AQI categories are imbalanced

### **6. Exploratory Data Analysis (EDA)**

* Univariate Analysis:  
  + Histograms for pollutant concentrations (PM2.5, PM10, NO2, etc.).
  + Boxplots to identify outliers and understand distribution.
  + Bar plots for categorical variables like AQI categories or stations.
* Bivariate/Multivariate Analysis:  
  + Correlation matrixUse sns.heatmap(df.corr(), annot=True) to identify relationships between variables
  + Visualize correlations between features
* Insights Summary:  
  + Temperature and humidity show moderate influence on pollutant concentration.
  + Wind speed negatively correlates with AQI—stronger winds help disperse pollutants.

### **7. Feature Engineering**

* Distance from industrial zones, traffic density estimates, or urban/rural tags.
* Adjust pollutants for weather (e.g., PM2.5 adjusted for wind speed).
* Derive AQI category as a separate feature (e.g., Good, Moderate, Unhealthy) if modeling pollutant levels.
* Ratios like PM2.5 / PM10 – indicates the proportion of fine vs coarse particles.
* Lag features: Previous hour/day values of pollutants (e.g., PM2.5\_t-1, AQI\_t-1).

### **8. Model Building**

* Choose the best model based on validation performance and generalization.
* Use Feature Importance (e.g., from Random Forest or XGBoost).
* Regression: Predict AQI or pollutant concentrations (PM2.5, PM10, etc.).
* Chronological split for time-series data (e.g., 70% train, 30% test).
  + Save the model using joblib or pickle.
  + Use SHAP values or LIME for local/global interpretability.

### **9. Visualization of Results & Model Insights**

* Plot actual vs. predicted values over time to assess temporal accuracy.
* Histogram of prediction errors to evaluate if they follow a normal distribution.
* Plot of errors (residuals) vs. predicted values to check for bias or patterns
* Line or scatter plot to show how well predictions follow actual AQI or pollutant levels.

### 

### **10. Tools and Technologies Used**

* Programming Language: Python.
* IDE/Notebook: Google Colab, Jupyter Notebook, VS Code, etc.
* Libraries: pandas, numpy, seaborn, matplotlib, scikit-learn, XGBoost, etc.
* Visualization Tools: matplotlib,seaborn,ploty,missingno

### 

### **11. Team Members and contributions**

* + Data cleaning:Ponmozhi P

Identified missing data and performed imputation techniques such as forward/backward fill for time-series data and mean/median imputation for stable features.

* + EDA:Subhashini.K

Conducted histograms, box plots, and bar plots to analyze the distribution of individual features (e.g., pollutant levels, temperature, wind speed).

* + Feature engineering:Subiksha.S  
     Created lag features to capture past pollutant concentrations (e.g., PM2.5 at time t-1) to help the model account for historical dependencies.
  + Model development:Srimathi.R  
     Researched and selected appropriate machine learning models based on the problem (regression for continuous AQI prediction or classification for AQI category prediction).