Phase-1 Submission

Student Name: RAKESH S

Register Number: 410723104061

Institution: DHANALAKSHMI COLLEGE OF ENGINEERING

Department: COMPUTER SCIENCE AND ENGINEERING

Date of Submission: 30.04.2025

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

1.Problem Statement

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

2.Objectives of the Project

The primary objective of this project is to develop and implement advanced machine learning algorithms to accurately predict air quality levels based on environmental, meteorological, and urban activity data. By leveraging predictive models, this project aims to provide timely and actionable insights into air pollution trends, enabling better decision-making for public health, urban planning, and environmental policy. Ultimately, the goal is to contribute to proactive environmental monitoring and enhance the overall understanding of air quality dynamics.

3.Scope of the Project

This project focuses on the development, training, and deployment of machine learning models to predict air quality levels in urban and semi-urban areas. The scope includes the following key areas:

* **Data Collection and Preprocessing**
* **Model Development**
* **Model Evaluation and Optimization**
* **Visualization and Insights**
* **Deployment and Use Cases**

4.Data Sources

Air Quality Data Sources:

Source	Description
Open AQ	**Aggregates air quality data from government and research stations worldwide**
U.S. EPA Air Now	**Real-time and historical AQI and pollutant data (mainly US)**
World Air Quality Index (WAQI)	**Global air quality API, city-level data**
Central Pollution Control Board (CPCB, India)	**Indian national air quality data**
European Environment Agency (EEA)	**EU countries' air quality data and APIs**

5.High-Level Methodology

1. Problem Definition & Objective Setting

- Define the target variable: e.g., PM2.5, AQI, or other pollutant levels.
- Set clear goals: e.g., short-term forecasting (hourly/daily), hotspot detection, or health advisory prediction.

---

2. Data Collection

- Gather multi-source datasets from:
  - Environmental sensors (e.g., Open AQ, WAQI)
  - Weather data APIs (e.g., Open Weather Map)
  - Traffic/urban activity (e.g., Google Maps, city portals)
  - Satellite imagery (e.g., NASA, Copernicus)

---

3. Data Preprocessing

- Handle missing values, outliers, and noise.
- Normalize/standardize numerical values.
- Parse and align timestamps (crucial for time-series modelling).
- Perform feature engineering:
  - Time-based features (hour, weekday, season)
  - Lag features (e.g., PM2.5 of last 1–3 hours)
  - Weather-traffic interactions (e.g., high traffic + low wind)

---

4. Exploratory Dat a Analysis (EDA)

- Visualize air pollutant trends across time and regions.
- Study correlation between pollutants, weather, and traffic.
- Identify pollution hotspots and seasonality.

---

5. Model Selection & Training

- Choose appropriate models based on the nature of your data:
  - **Tree-based models:** Random Forest, XG Boost (good for tabular data)
  - **Time-series models:** ARIMA, LSTM, GRU (good for sequential forecasting)

  - **Hybrid models:** Combine deep learning + classic ML features
- Split data into training, validation, and testing sets.
- Use cross-validation to ensure model robustness.

---

6. Model Evaluation

- Use evaluation metrics like:
  - **MAE (Mean Absolute Error)**
  - **RMSE (Root Mean Square Error)**
  - **R^2 Score**
- Compare model predictions with ground truth.
- Visualize actual vs predicted pollutant trends.

---

7. Deployment (Optional but Recommended)

- Deploy as a web or mobile dashboard using:
  - Flask/Django (backend)
  - Stream lit or Dash for visual insights
- Integrate real-time predictions using live data APIs.

---

8. Insights & Recommendations

- Identify pollution spikes and their causes (e.g., weather, traffic).
- Generate visual reports for public health authorities or environmental researchers.
- Provide early warnings or health alerts.

6.Tools and Technologies

Programming Languages

- **Python** – Primary language for data processing, modeling, and deployment
  - Widely supported in machine learning, data analysis, and API interaction

---

Libraries & Frameworks

Data Handling & Analysis

- **Pandas** – For structured data manipulation

- **NumPy** – For numerical computations
- **Matplotlib / Seaborn / Plotly** – For visualizations and EDA

Machine Learning & Deep Learning

- **Scikit-learn** – For traditional ML models (e.g., Random Forest, XGBoost)
- **XGBoost / LightGBM** – High-performance gradient boosting
- **TensorFlow / Keras** – For deep learning (e.g., LSTM, GRU models)
- **PyTorch** – Alternative deep learning library (especially for time-series)

Time-Series Analysis

- **Statsmodels** – For ARIMA and other statistical models
- **Prophet (by Facebook)** – Easy-to-use tool for time-series forecasting
- **TSFresh** – For automated time-series feature extraction

APIs & Data Access

- **OpenAQ API / WAQI API** – For air quality data
- **OpenWeatherMap API / WeatherStack** – For meteorological data
- **Google Maps API / TomTom API** – For traffic data
- **NASA EarthData / Google Earth Engine** – For satellite-based data

---

Data Storage

- **CSV / Excel** – Simple file-based storage
- **SQL / SQLite** – For structured and queryable data
- **MongoDB** – NoSQL database (for semi-structured sensor data)
- **AWS S3 / Google Cloud Storage** – Cloud-based storage for large datasets

---

Visualization & Reporting

- **Power BI / Tableau / Google Data Studio** – For creating dashboards (optional)
- **Streamlit / Dash** – For creating lightweight, interactive web apps

---

Model Deployment

- **Flask / Django** – Backend frameworks to host models
- **Streamlit / Dash** – For quick deployment with visualization
- **Docker** – Containerize your model for scalable deployment
- **AWS / Heroku / Google Cloud / Azure** – Cloud deployment platforms

Model Experiment Tracking (Optional but Recommended)

- **MLflow** – For tracking experiments, models, and metrics
- **Weights & Biases** – Real-time logging of training and hyperparameter tuning

7.Team Members and Roles

NAMES	ROLES	RESPONSBILITY
NITHIN CHANDER R	LEADER	DATA COLLECTION AND DATA CLEANING
VISHWANATHAN P	MEMEBER	EXPLORATORY DATA ANALYSIS (EDA)
SANTHOSH S	MEMBER	TOOLS AND TECHNOLOGIES
RAKESH S	MEMBER	VISUALIZATION AND INSIGHTS
SAJIT KUMAR E	MEMBER	MODEL EVALUATION AND OPTIMIZATION