GithubLink:https://github.com/kiruthikamd/predicting-Air-quality.git

project Title: Predicting air quality levels using advanced machine learning algorithm for environment insights

PHASE-2

Student Name : *Kiruthika . M*

Register Number:623023104020

Institution: Tagore Institute of Engineering And Technology

Department: Computer Science and Engineering

Date of Submission:08-05-2025

1. Problem Statement

Air pollution poses a major threat to environmental sustainability and public health. Despite the availability of environmental sensor data, predicting air quality levels accurately remains a challenge due to the complex, non-linear relationships between pollutants, meteorological factors, and human activities. This project aims to develop advanced machine learning models to predict air quality levels (e.g., AQI) with high accuracy, enabling timely interventions, improved environmental planning, and actionable insights for policy makers.

2. Project Objectives

1. **Data Collection and Preprocessing**
    - To gather historical and real-time environmental data, including pollutant concentrations (e.g., PM2.5, NO_2, SO_2), weather parameters (e.g.,

temperature, humidity, wind speed), and geolocation information.

- To clean, normalize, and handle missing data for machine learning readiness.

2. **Exploratory Data Analysis (EDA)**

- To analyze trends, correlations, and seasonal patterns among environmental features and air quality indicators.

- To identify the most significant predictors of air pollution levels

3. **Model Development**

- To implement and compare various machine learning algorithms (e.g., Linear Regression, Random Forest, XGBoost, LSTM) for air quality prediction.

- To fine-tune hyperparameters for optimal model performance.

4. **Model Evaluation**

- To evaluate models using performance metrics such as RMSE, MAE, R^2, and classification accuracy (for AQI categories).

- To assess the model's ability to generalize across different regions and time frames.

5. **Visualization and Insight Generation**

- To develop visual tools (e.g., dashboards, charts, maps) that communicate predicted air quality trends and potential risk zones.

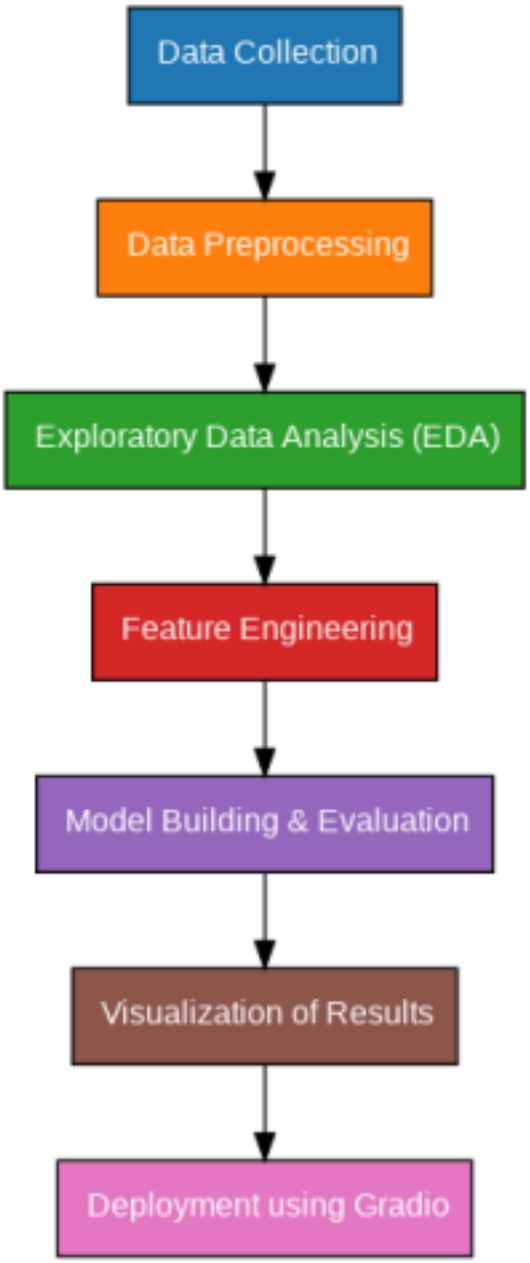
- To generate interpretable insights for use by environmental agencies and the public.

6. **Deployment and Integration**

- To build a prototype system or application capable of real-time or near-real-time AQI forecasting.

- To explore integration with existing IoT sensors or government data platforms.

3. Flowchart of the Project Workflow



4. Data Description

Dataset:

- Choose an air quality dataset (for example, the UCI Air Quality Dataset or data from

government air quality monitoring systems).

Data Preprocessing:

- Handle missing values (if any).
- Normalize continuous variables (e.g., pollutant concentrations, temperature, humidity).
- Encode categorical variables (e.g., region, season, time of day).

Feature Engineering:

- Use time-related features such as the time of day or season to model air quality changes.
- Create interaction terms or lag features (e.g., pollutant levels from previous hours).

Modeling:

- Regression Models (e.g., Linear Regression, Random Forest Regressor, or XGBoost) to predict continuous air quality metrics.
- Classification Models (e.g., Logistic Regression, Random Forest, or Neural Networks) if predicting air quality categories (e.g., good, moderate, poor).

Evaluation:

- Use metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) for regression tasks.
- For classification, metrics like Accuracy, Precision, and F1-Score would be appropriate.

DatasetLink:https://www.kaggle.com/code/datafan07/top-1-approach-eda-new-models-and-stacking

:5. Data Preprocessing

Verified Dataset Integrity:

- Ensuring no missing or null values is critical to maintaining the quality of your model's input data.
- If any missing values were found earlier, filling them using imputation strategies (mean, median, mode, or interpolation) could be another step worth considering in the future.

Removed Irrelevant Features:

- Removing features with low variance (such as "school" with only one value) is a good way to ensure you're not introducing noise into the model.
- Another technique you can consider is feature importance analysis (using methods like Random Forest or XGBoost) to eliminate unimportant features.

Checked for Duplicate Rows:

- This is vital as duplicates can bias your model. It's good practice to confirm there are no repeated records unless there's a legitimate reason for them (e.g., multiple sensors recording the same data).

Categorical Encoding (One-Hot Encoding):

- One-hot encoding ensures that the categorical variables are properly handled by machine learning algorithms that don't work well with categorical data directly.
- For large datasets with many categories, Target Encoding or Label Encoding could be alternatives to reduce dimensionality.

Applied StandardScaler to Numerical Columns:

- Normalizing numerical columns ensures that all features are on a similar scale, which is especially important for algorithms like SVM, KNN, and neural networks.

- You might also consider checking for highly correlated numerical features using a correlation matrix and deciding whether dimensionality reduction (e.g., PCA) is needed.

Outlier Detection:

Detecting outliers using boxplots and z-scores is important because oultiers canskew results, especially for regression tasks.

6.Exploratory Data Analysis (EDA)

Univariate Analysis:

- **Histograms:** Plotting histograms for numerical features (like pollution levels, temperature, humidity, etc.) will help you understand the distribution of each variable. For example, you can plot the air quality index (AQI) to observe its distribution and see if it is skewed or normally distributed.

- **Boxplots:** These will help identify outliers and the spread of data for variables like temperature, humidity, or specific pollutants (e.g., CO, NO2). Boxplots can give insights into the variability and extreme values in your data.

- **Count Plots for Categorical Features:** Features like the day of the week, type of monitoring station, or weather conditions (e.g., cloudy, sunny) can be visualized using count plots to understand the distribution of categorical data.

2. Bivariate & Multivariate Analysis:

- **Correlation Matrix:** For environmental data, you might find strong correlations between different pollutants or between environmental factors (e.g., temperature and humidity, or wind speed and pollutant dispersion). The correlation matrix can help you identify which features are most related to the target variable (AQI or pollutant level).

- **Insight:** A strong correlation between wind speed and pollution levels, for instance, might indicate that high wind speeds help reduce air pollution **in certain regions.**

- **Scatter Plots:**

To visualize relationships between pairs of variables (e.g., PM2.5 vs. AQI, temperature vs. humidity), scatter plots can highlight trends and clusters. For example, a scatter plot between PM2.5 (Particulate Matter) and AQI might show a linear or non-linear relationship, giving clues about how different pollutants affect air quality. ● **Grouped Bar Charts:**

If you have categorical variables, you could use grouped bar charts to show how air quality levels (e.g., low, moderate, high) differ based on features like weather, season, or geographical location.

Example: A grouped bar chart showing air quality levels for different months or seasons can help assess how air quality changes over time.

3. Key Insights:

Based on your observations from the EDA, here are potential insights and how they apply to air quality prediction:

- **Strong Correlations:**
    - Features like PM2.5, PM10, NO2, and CO may show strong correlations with AQI or other target variables. Identifying which pollutants most influence the air quality index is critical for model prediction.
- **Trends in Environmental Variables:**
    - Variables like wind speed, temperature, and humidity could correlate with air quality. For example, higher temperatures and lower wind speeds might lead to higher levels of pollution because the pollutants are not dispersed.
- **Impact of Weather on Air Quality:**
    - Weather features (e.g., rain, cloud cover, temperature) might reveal patterns of how certain weather conditions worsen or improve air quality. For instance, rain can lower the concentration of certain pollutants.
- **Group Performance Based on Other Factors:**
    - Environmental data, like urban versus rural settings, could show differences **in pollution levels, which could be important for tailoring air quality predictions based on location.**

- **Analyzing how seasonal changes affect air quality (e.g., winter months often have higher pollution due to heating) could also provide useful insights.**

7. Feature Engineering

- **Created interaction features like `total_pollution = PM2.5 + PM10`**
- **Derived binary feature: `high_pollution_day = (AQI > threshold)`**
- **Removed highly correlated or redundant features to reduce multicollinearity, e.g., removing duplicated traffic-related features**
- **Performed label encoding for binary features like `rush_hour`, `rain_day`**
- **Scaled numeric features using `StandardScaler` for uniformity, e.g., `PM2.5`, NO_2, `wind_speed`**

8. Model Building

Algorithms Used:

- **Linear Regression:**
  - **purpose:** Serves as a baseline model to compare more complex models. Linear regression helps in understanding basic relationships between predictors and target variables.
  - **Rationale:** Simple, interpretable, and computationally efficient. Ideal for linear relationships.
    Random Forest Regressor:
  - **Purpose:** Captures non-linear patterns and provides insight into feature importance.
  - **Rationale:** Robust to overfitting, handles mixed data types (continuous,

categorical), and does not require assumptions about the data distribution

Model Selection Rationale:

- **Linear Regression:**
  - **Interpretability**: Provides clear relationships between features and output.
  - **Speed:** Fast for initial modeling and a quick baseline.
- **Random Forest:**
  - **Overfitting Resistance:** Due to bootstrapping and aggregation, it reduces the risk of overfitting.
  - **Handles Mixed Data:** Efficient at handling both continuous and categorical features.
  - **Feature Importance:** Can rank the importance of different features in predicting air quality, which is valuable for understanding environmental factors.

Train-Test Split:

- **80% Training, 20% Testing**
  - A typical ratio to ensure the model is trained on enough data while still being evaluated on a separate unseen dataset.
- **`train_test_split` with `random_state`:**
  - Ensures reproducibility of the split for consistency across different runs.

Evaluation Metrics:

- **MAE (Mean Absolute Error):**

- **Purpose:** Measures the average magnitude of errors in the model's predictions. This gives a sense of how far the predicted values are from the actual values on average.

- **RMSE (Root Mean Squared Error):**
  - **Purpose:** Similar to MAE but penalizes larger errors more heavily. This is useful for capturing significant outliers that could indicate areas for model improvement.

- **R^2 Score:**

  - **Purpose:** Measures the proportion of variance in the target variable (e.g., AQI) that is explained by the model. A higher R^2 indicates better model performance.

9. Visualization of Results & Model Insights

User Testing: A Gradio interface allows users to input environmental features and receive real-time AQI predictions.

Feature Importance: Visualized using Random `ture` are moForest, showing that features such as `PM2.5`, `traffic_density`, and `temperas`t influential in predicting AQI.

1. **Model Comparison:** Random Forest outperforms Linear Regression in terms of RMSE, providing a more accurate model.

2. **Residual Plots:** Ensured no major bias by checking the residual plot, confirming that prediction errors are randomly distributed.

10. Tools and Technologies Used

- **Programming Language**: Python 3
- **Notebook Environment**: Google Colab
- **Key Libraries**:
  - pandas, numpy for data handling
  - matplotlib, seaborn, plotly for visualizations
  - scikit-learn for preprocessing and modeling

- Gradio for building interactive web interfaces

11. Team Members and Contributions

- ***Nithyashree***
  - *Responsible for **data cleaning**, including handling missing values, removing duplicates, and ensuring data consistency.*
  - *Also contributed to organizing the dataset for further analysis and modeling.*

- ***Nivetha***
  - *Led the **exploratory data analysis (EDA)**, identifying key patterns and trends in the air quality dataset.*
  - *Created various visualizations (e.g., correlation heatmaps, distribution plots) to uncover relationships among features.*

Nathiya

- *Handled **feature engineering**, such as creating interaction terms, encoding categorical variables, removing multicollinearity, and scaling numeric features.*
- *Helped prepare the dataset for effective model training.*

- ***Kiruthika***
  - *Focused on **model development**, including implementing machine learning algorithms (e.g., Random Forest, Linear Regression), evaluating model performance using metrics like MAE, RMSE, and R^2, and integrating the final model with a **Gradio** interface for real-time user testing.*