

Phase-3 Submission

Student Name: GUKAPRIYA M

Register Number:731323104014

Institution: JKKN COLLEGE OF ENGINEERING & TECHNOLOGY

Department: BE-CSE

Date of Submission: 16.05.2025

Github Repository Link: <https://github.com/Infosanjay-2k04/Batch-10.git>

1. Problem Statement

Air pollution poses a significant threat to public health and the environment, especially in rapidly urbanizing regions. Rising levels of pollutants such as PM_{2.5}, NO₂, CO, and O₃ contribute to respiratory diseases, environmental degradation, and reduced quality of life. Accurate prediction of air quality levels is essential for timely health advisories, effective pollution control, and informed policy-making.

2. Abstract

Air pollution is a growing concern in urban areas, adversely affecting human health and the environment. The objective of this project is to develop a predictive model that accurately forecasts air quality levels using historical pollution and weather data. We focus on predicting either the Air Quality Index (AQI) or individual pollutant concentrations such as PM_{2.5} and NO₂. The problem is approached using advanced machine learning algorithms, including regression and classification techniques, depending on the target variable.

3. System Requirements

Hardware Requirements

- **RAM:** Minimum 8 GB (16 GB recommended for large datasets or training complex models)
- **Processor:** Intel i5 (8th Gen or above) or equivalent; GPU recommended for deep learning models
- **Storage:** At least 5 GB of free disk space for datasets and model files

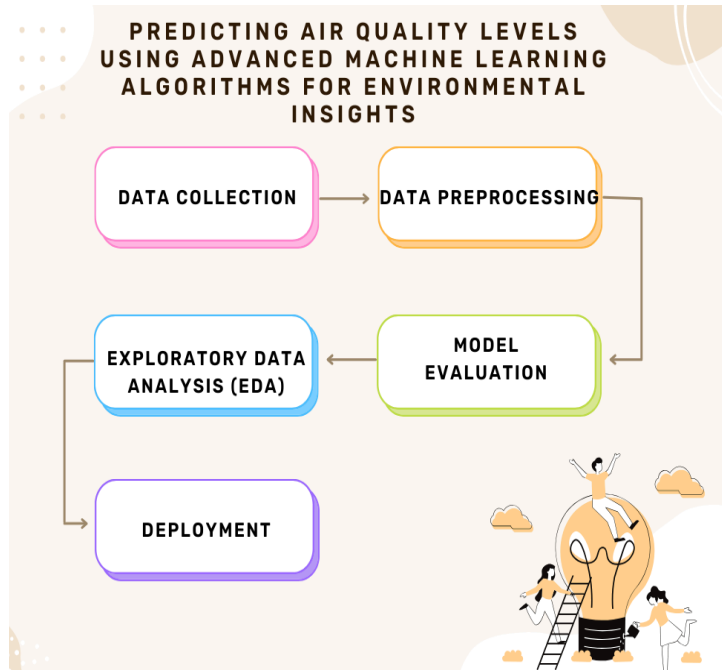
Software Requirements

- **Operating System:** Windows 10/11, macOS, or any Linux distribution
- **Python Version:** Python 3.8 or above
- **IDE/Environment:** Jupyter Notebook, Google Colab, or VS Code with Jupyter extension

4. Objectives

- Classify air quality into standard categories such as *Good*, *Moderate*, or *Unhealthy* (if using classification models).
- Generate **early warnings** about poor air quality conditions.
- Provide **actionable insights** into how environmental variables influence pollution levels.
- Enable **data-driven decision-making** for public health agencies and urban planners.
-

5. Flowchart of Project Workflow



6. Dataset Description

Source: The dataset used for this project is obtained from **Kaggle**, titled *"Air Quality Data in India"*. It contains air pollution records collected from various cities across India, including data from the Central Pollution Control Board (CPCB).

Type: Public

Access Link: <https://www.kaggle.com/datasets/rohanrao/air-quality-data-in-india>

Size and Structure:

Rows: ~29,000

Columns: 15

Key columns include:

City, Date, PM2.5, PM10, NO, NO2, NOx, CO, SO2, O3, Benzene, Toluene, AQI, AQI_Bucket

Sample Data (df.head()):

Date	P	P	I	N	N	(S	A	AQI_Bucket
M	M	(O	O	(O	Q		

		2.5	10		2	x		2	1	
0	2015-01-01	19	24	1	4	4	(1	2	Poor
	1	8.	7	4	2	9	.	2	7	
		0					1		9	
1	2015-01-01	74	10	1	2	2	(1	1	Moderate
	1	.0	3	1	2	8	.	0	4	
							1		2	
2	2015-01-01	56	88	1	1	2	(7	1	Moderate
	1	.0		1	8	1	.		1	
							1		7	
3	2015-01-01	89	12	1	3	3	(1	1	Moderate
	1	.0	2	1	0	6	.	1	6	
							1		8	
4	2015-01-01	42	68	1	1	1	(6	9	Satisfactory
	1	.0			4	7	.		3	
							1			

7. Data Preprocessing

1. Handling Missing Values
2. Removing Duplicates
3. Handling Outliers
4. Feature Encoding
5. Feature Scaling

8. Exploratory Data Analysis (EDA)

1. Histograms

- **Purpose:** Show distribution of pollutants like **PM2.5**, **PM10**, **N02**.
- **Insight:** Most pollutants are right-skewed, indicating occasional high pollution spikes.

2. Boxplots

- **Purpose:** Identify outliers and compare pollutant levels across cities or AQI categories.
- **Insight:** Cities like Delhi show consistently higher outliers in **PM2.5** and **N02**.

3. Heatmap (Correlation Matrix)

- **Purpose:** Reveal correlations between features.
- **Insight:** Strong positive correlation observed between **PM2.5**, **PM10**, and **AQI**.

4. Line Plot (Trend Over Time)

- **Purpose:** Visualize AQI or pollutant levels over months/years.
- **Insight:** AQI peaks during winter months, possibly due to stagnant air and festival-related emissions.

5. Pair Plot

- **Purpose:** Visualize interactions between multiple pollutant variables.
- **Insight:** Clusters emerge in pollution patterns, supporting classification potential.

9. Feature Engineering

1. New Feature Creation
2. Feature Selection
3. Transformation Techniques

10. Model Building

1. Baseline Models
2. Advanced Models

Model Evaluation Metrics Used

- **Regression:**
 - RMSE (Root Mean Squared Error)
 - MAE (Mean Absolute Error)
 - R^2 Score
- **Classification:**
 - Accuracy
 - Precision, Recall, F1-score
 - Confusion Matrix

11. Model Evaluation

Accuracy

- Percentage of correctly predicted labels over all predictions.
- Good for balanced classes.

Precision

- Measures how many predicted positive cases were actually positive.
- Useful when false positives are costly.

12. Deployment

Steps Followed for Deployment:

1. Built a Streamlit App
2. Saved the Trained Model
3. Created a GitHub Repository

Files Used in Deployment:

1. **streamlit_app.py**: main app interface
2. **aqi_model.pkl**: serialized machine learning model

3. **requirements.txt**: Python packages (e.g., **pandas**, **scikit-learn**, **streamlit**)

13. Source code

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import joblib

# Load data
df = pd.read_csv("air_quality_data.csv")

# Preprocessing
df.dropna(inplace=True)
df = df[['PM2.5', 'PM10', 'NO2', 'SO2', 'CO', 'O3', 'AQI']]

# Feature and target
X = df.drop('AQI', axis=1)
y = df['AQI']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Model
model = RandomForestRegressor()
model.fit(X_train, y_train)

# Evaluate
y_pred = model.predict(X_test)
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))
print("MAE:", mean_absolute_error(y_test, y_pred))
print("R² Score:", r2_score(y_test, y_pred))

# Save model
joblib.dump(model, "aqi_model.pkl")
```

```
X_test_scaled = scaler.transform(X_test)
```

```
# Step 7: Train a model (Random Forest)
```

```
model = RandomForestClassifier(n_estimators=100, random_state=42)
```

```
model.fit(X_train_scaled, y_train)
```

```
# Step 8: Make predictions
```

```
y_pred = model.predict(X_test_scaled)
```

```
# Step 9: Evaluation
```

```
print("Classification
```

```
Report:\n")
```

```
print(classification_report(y_test, y_pred))
```

```
# Confusion matrix
```

```
cm = confusion_matrix(y_test, y_pred)
```

```
plt.figure(figsize=(8,6))
```

```
sns.heatmap(cm, annot=True, fmt='d',
```

```
cmap='Blues') plt.xlabel("Predicted")
```

```
plt.ylabel("True")
```

```
plt.title("Confusion
```

```
Matrix") plt.show()
```

14. Future scope



1. Integration of Real-Time Data via APIs

- **What:** Connect to live environmental monitoring APIs (e.g., OpenAQ, AQICN).
- **Why:** Enables real-time AQI predictions, making the model more practical and dynamic.

- **Benefit:** Useful for city administrations, environmental apps, and alert systems.
-



2. Incorporation of Weather and Geographic Features

- **What:** Add features like temperature, humidity, wind speed, elevation, and urban density.
- **Why:** These factors significantly influence pollutant dispersion and AQI.
- **Benefit:** Enhances model accuracy and captures seasonal or regional variations more effectively.

13. Team Members and Roles

1. M.SIVARANJANI – Project Lead & Data Analyst

- Defined the project scope and objectives
- Collected and cleaned the dataset
- Performed exploratory data analysis (EDA)
- Created data visualizations and initial insights

2. M.GUKAPRIYA– Machine Learning Engineer






- Handled data preprocessing and feature engineering
- Built and tuned machine learning models (Random Forest, XGBoost, etc.)
- Evaluated models using regression and classification metrics
- Saved and exported trained models using `joblib`

3. K.PRABHAKARAN – Deployment & Frontend Developer




- Designed and developed the Streamlit web app interface
- Integrated the model with a user-friendly form for AQI predictions
- Created the `requirements.txt` and managed deployment on Streamlit Cloud
- Assisted in writing documentation and creating a demo

4.V.JAGAN – Documentation & Reporting

- Compiled all code and results into a structured project report
- Drafted the abstract, problem statement, objectives, and future scope
- Created visuals for flowcharts and screenshots of project stages
- Managed version control and GitHub repository

Gukapriya /
Batch-11

[Code](#) [Issues](#) [Pull requests](#) [...](#)





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

☆ 0 stars 🍴 0 forks 👁 1 watching
🌿 1 Branch 🏷 0 Tags 📈 Activity
🌐 Public repository


🌿 main ▾

Code ▾

...

**Gukapriya** 7 minutes ago ... 

📄 README.md	22 minutes ago
📄 Source code	15 minutes ago
📄 output.pdf	7 minutes ago

 **README**

