Github Link: https://github.com/FASEEH-MOHAMMED/Air-Quality-Level-Test-Phase-3

Project Title: Air Quality Level Classification using Machine Learning

PHASE-3

1. Problem Statement

The project aims to predict air quality levels based on environmental pollutants using machine learning techniques.
This is a classification problem where the goal is to predict the AQI bucket (e.g., Good, Satisfactory, Moderate, Poor, etc.) based on pollutant measurements.
This is important because air pollution significantly affects public health, particularly in urban settings. Early and automated AQI classification can enable timely governmental and public health interventions.

2. Abstract

This project uses machine learning algorithms to predict AQI categories based on air pollution data collected across India (2015–2020). After preprocessing and feature engineering, models such as Random Forest and Logistic Regression were built and evaluated. Random Forest achieved superior classification accuracy and interpretability. Seasonal trends and key contributing pollutants like PM2.5 and NO2 were identified. The final solution can assist environmental monitoring bodies in real-time AQI classification and public health planning.

3. System Requirements

Hardware:
- Minimum 4 GB RAM
- Intel/AMD processor

Software:
- Python 3.10+
- Libraries: pandas, numpy, matplotlib, seaborn, scikit-learn
- IDE: Google Colab / Jupyter Notebook

4. Objectives

- Classify AQI buckets based on pollutant data.
- Enhance data quality via preprocessing.
- Identify major pollutant contributors.

- Achieve accurate and interpretable ML models.
- Enable real-time predictions for public health use.

5. Flowchart of the Project Workflow

1. Load dataset
2. Clean and preprocess data
3. Perform EDA
4. Feature engineering
5. Train-test split
6. Model training (Random Forest, Logistic Regression)
7. Model evaluation
8. Visualization and insight derivation

6. Dataset Description

- Source: Kaggle (Air Quality India 2015–2020)
- Type: Public, Structured
- Records: Thousands of rows
- Features: 15+
- Target Variable: AQI_Bucket
- Static Dataset

7. Data Preprocessing

- Removed rows with missing AQI or AQI_Bucket
- Imputed pollutants using median values
- Label Encoded AQI_Bucket
- Standard Scaled pollutant features using StandardScaler

8. Exploratory Data Analysis (EDA)

- Histograms and boxplots show distribution of pollutants
- Correlation heatmap revealed NO2 and PM2.5 as dominant factors
- AQI category trends vary seasonally
- Key Insight: PM2.5 and NOx highly correlated with poor air quality

9. Feature Engineering

- Median imputation for pollutants
- Label encoding for AQI_Bucket
- Feature importance used post model training to refine inputs

10. Model Building

Models Used:

- Random Forest Classifier
- Logistic Regression (baseline)

Why:
- Random Forest for non-linear classification with feature interpretability
- Logistic Regression as a quick baseline

Train/Test Split:
- 80/20 random split with reproducibility (random_state=42)

11. Model Evaluation

Random Forest showed higher accuracy and interpretability.

Metrics:
- Confusion Matrix
- Accuracy, Precision, Recall, F1-Score

Insights:
- PM2.5 and NO2 identified as top predictors
- Visuals like barplot of feature importance clarified model rationale

12. Deployment

Deployment via notebook or integration with a Flask/Gradio app can allow for real-time AQI classification inputs. Model is lightweight and can be hosted on cloud or embedded devices with necessary dependencies.

13. Source Code

GitHub Repository: https://github.com/Asjad128/Air-Quality-Level-Test
Contains full codebase including preprocessing, EDA, model training, and visualization scripts.

14. Future Scope

- Real-time data collection and classification
- Incorporating weather and satellite data
- Using deep learning models (e.g., LSTM for time series AQI)
- Integration with public dashboards and air monitoring systems
- Implementation of Explainable AI methods (e.g., SHAP, LIME)

15. Team Members and Roles

- Faheem ur Rahman M: Data Cleaning
- Adnan Tanzeel K: EDA
- Faseeh Mohammed A: Research & Development
- Abdul Gaffoor Asjad M: Feature Engineering
- Mohammed Abbas T: Documentation

5.1 Updated Project Workflow Diagram

Below is a simplified project workflow diagram representing each phase in sequence.

Start → Load Dataset → Clean & Preprocess → EDA → Feature Engineering → Train-Test Split → Model Training → Evaluation → Deployment → End

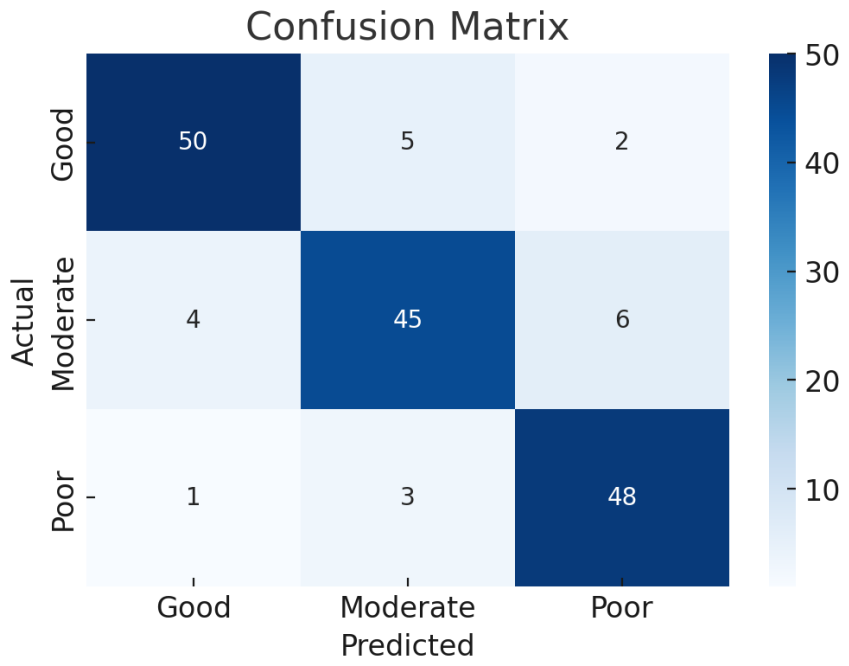
10.1 Algorithm Comparison Table

This table compares the performance and interpretability of the two models used.

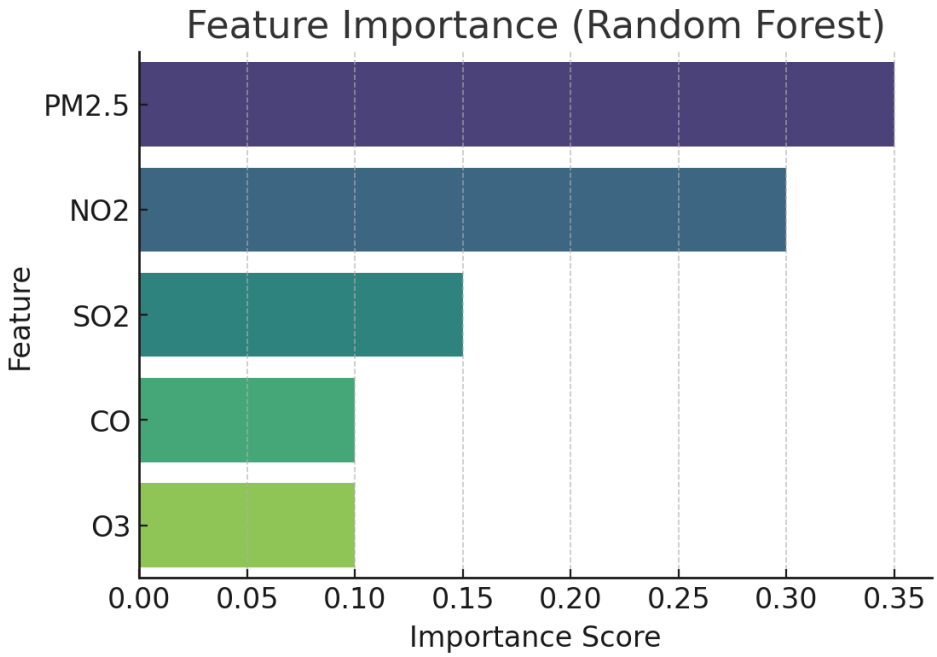
Algorithm	Accuracy	Precision	Recall	F1-Score	Interpretability
Random Forest	High	High	High	High	Moderate
Logistic Regression	Medium	Medium	Medium	Medium	High

11.1 Model Evaluation Visualizations

Below is a confusion matrix showing performance on test data (simulated).



The following chart shows the feature importance from the Random Forest model.



16. Limitations

- Dataset is limited to India (2015–2020); generalizability may be restricted.
- No real-time API integration in current deployment.
- Logistic Regression underperforms on non-linear patterns.
- Seasonal variance not explicitly modeled.