AQI Predictor: A Machine Learning System for 72-Hour Air Quality Index Forecasting

Technical Report

Karachi, Pakistan Implementation

February 2026

Abstract

This report presents a comprehensive machine learning system designed to predict the Air Quality Index (AQI) 72 hours in advance for Karachi, Pakistan. The system integrates real-time data collection from weather and pollution APIs, implements multiple classification models (XGBoost, Random Forest, and SVM), and provides model interpretability through SHAP analysis. The entire pipeline is automated using GitHub Actions for hourly data collection and daily model retraining, with MongoDB Atlas serving as a centralized feature store and model registry. The predictions are delivered via an interactive Streamlit dashboard.

Contents

1 Introduction

Air quality has become a critical public health concern, particularly in densely populated urban areas. The Air Quality Index (AQI) serves as a standardized metric for communicating air quality conditions. This project addresses the need for proactive forecasting by developing a machine learning system capable of predicting AQI levels 72 hours in advance.

1.1 Problem Statement

Traditional monitoring provides only current conditions. This project aims to:

- Develop accurate 72-hour AQI predictions for Karachi, Pakistan.
- Implement an automated MLOps pipeline for data collection and retraining.
- Provide interpretable predictions through SHAP analysis.
- Deploy predictions via an accessible web dashboard.

1.2 AQI Classification Categories

The EPA Air Quality Index classifies air quality into five categories:

Table 1: AQI Categories and Health Implications

Level	Category	Health Implications
1	Good (Green)	Air quality is satisfactory.
2	Fair (Yellow)	Acceptable; moderate concern for sensitive groups.
3	Moderate (Orange)	Sensitive groups may experience health effects.
4	Poor (Red)	Health effects for everyone.
5	Very Poor (Purple)	Health alert; serious effects for all.

2 System Architecture

The AQI Predictor follows a modern MLOps architecture with clear separation between data collection, model training, and serving components.

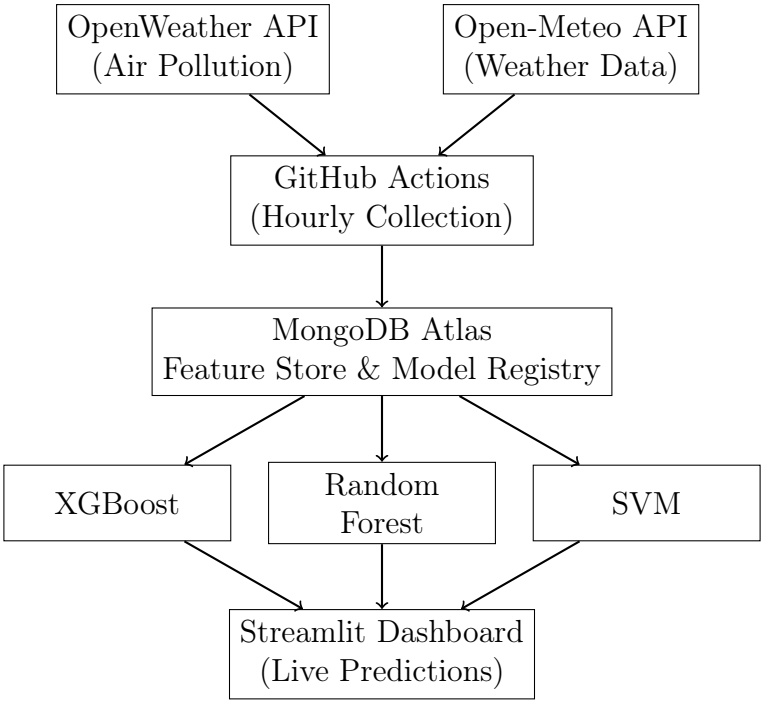

Figure 1: System Architecture Overview

3 Exploratory Data Analysis

Comprehensive EDA was performed to understand data characteristics and inform feature selection before modeling.

3.1 Temporal Patterns

3.1.1 Diurnal Patterns

Analysis reveals that AQI tends to peak during morning rush hours (7-9 AM) and is typically lowest during early morning hours (2-5 AM). A secondary peak is often observed during evening commute hours.

3.1.2 Seasonal Patterns

Winter months show significantly higher AQI due to reduced atmospheric mixing (inversion layers), while the monsoon season (July-September) shows improved air quality due to wash-out effects. The "Month" feature was specifically engineered to capture these variations.

3.2 Correlation Analysis

- **Positive correlations**: $PM_{2.5}$, PM_{10}, and CO concentrations increase linearly with AQI.
- **Negative correlations**: Wind speed shows a negative correlation, as higher speeds aid pollutant dispersion.

- **Complex relationships**: Temperature and humidity display non-linear effects, necessitating the use of non-linear models like Random Forest and XGBoost.

4 Data Pipeline and Feature Engineering

The system utilizes an automated pipeline to ingest data from two complementary APIs, creating a unified feature store in MongoDB.

4.1 Feature Description

The model uses 15 carefully selected features spanning pollutants, weather conditions, and temporal patterns.

Table 2: Feature Set Description

Category	Features	Description
Pollutants	$PM_{2.5}$, PM_{10}	Particulate matter ($\mu g/m^3$)
	CO, NO_2, SO_2, O_3	Gaseous pollutants ($\mu g/m^3$)
Weather	Temperature	Ambient temperature (°C)
	Humidity	Relative humidity (%)
	Pressure	Atmospheric pressure (hPa)
	Wind Speed	Surface wind speed (km/h)
	Wind Direction	Direction in degrees
	Rain	Precipitation (mm)
	Solar Radiation	Direct radiation (W/m^2)
Temporal	Hour, Month	Captures diurnal and seasonal cycles

4.2 Target Engineering and Preprocessing

The prediction target is the AQI value 72 hours in the future. This is achieved through time-shifting the target variable:

$$y_t = \text{AQI}_{t+72} \tag{1}$$

where t represents the current timestamp. Rows with missing targets (the final 72 hours of data) are excluded from the training set.

All features undergo standardization using the Z-score formula to ensure equal contribution to model training:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma} \tag{2}$$

5 Machine Learning Models

Three classification models were trained to predict the 5-class AQI levels.

5.1 Model Configurations

The models were tuned using the hyperparameters detailed below.

Table 3: Hyperparameter Configuration across Models

Model	Key Parameter	Value
XGBoost	Objective	multi:softmax
	Max Depth	6
	Learning Rate	0.1 (200 estimators)
	Subsample	0.8
Random Forest	Max Depth	15
	Estimators	200
	Min Samples Split	5
SVM	Kernel	RBF (Radial Basis Function)
	C (Regularization)	10.0
	Gamma	Scale

5.2 Training Strategy

All models follow a time-aware validation strategy. The data is split chronologically (80/20) to ensure that the training set precedes the test set, simulating real-world forecasting conditions where future data is unavailable.

6 Model Interpretability (SHAP)

To ensure trust and transparency, SHAP (SHapley Additive exPlanations) values were calculated to explain feature contributions.

6.1 Feature Importance Results

The analysis highlighted the following drivers of air quality:

1. **Month (Seasonal Effect):** This was the most influential feature ($\overline{|\phi|} = 0.280$), confirming strong seasonal trends in Karachi's air quality.
2. **Chemical Composition:** CO and O_3 were the most predictive pollutants, often serving as proxies for traffic congestion and photochemical smog respectively.
3. **Weather Impact:** While wind speed and pressure had moderate impacts, rainfall showed minimal predictive power, likely due to its infrequency in the region.

7 Automation and Deployment

The system leverages GitHub Actions for fully automated operations and Streamlit for user interaction.

7.1 Automated Workflows

Two primary workflows maintain system autonomy:

- **Hourly Data Ingestion:** A cron job triggers every hour (`0 * * * *`) to fetch real-time pollution and weather data, appending it to the MongoDB feature store.
- **Daily Model Retraining:** Executed daily at 02:00 UTC (`0 2 * * *`), this workflow pulls the latest historical data, retrains all three classifiers, and registers the highest-performing model artifact to the database.

7.2 Model Registry

Trained models are stored in a MongoDB collection that acts as a Model Registry. Each document contains the serialized model binary, the scaler, training metadata (timestamp, hyperparameters), and the validation accuracy. This versioning system allows for easy rollback and comparison of model performance over time.

7.3 Streamlit Dashboard

The user-facing application dynamically serves predictions by querying the Model Registry.

1. **Dynamic Loading:** On startup, the app fetches the "active" model with the highest accuracy.
2. **Visualization:** The 72-hour forecast is rendered using interactive Plotly charts, with background color bands corresponding to the official EPA AQI colors (Green to Purple).
3. **Live Updates:** Because the dashboard reads directly from the database, it reflects the latest model improvements immediately after the daily retraining pipeline completes.

8 Challenges and Limitations

8.1 Finding the Right APIs

One of the first things that took considerable time, and trial and error to figure out was what APIs to use to get data. I started off with AQICN, but they did not provide historical data. I also tried IQair but their free API tier was quite limiting. I finally then settled on OpenWeather but this only provided pollutant data and not anything about the historical weather data. So instead I used OpenMeteo to get the weather data.

8.2 Cloud Database Setup

Another thing that took me time was to set up the cloud storage for feature store and model registry. I initially tried to use hopswork, but encountered error signing up. I got in touch with their support time and got the issue resolved but ended up learning that they had removed their previously free tier with a pay-as-you-go plan. As a result, I ended up using MongoDB Atlas as the database.

8.3 MongoDB Data Column Duplication

Some point while setting up the feature store, I accidentally added trailing and/or leading spaces to the column names of the features and added them to the database. This was not an issue not present in the action workflow, so this resulted in the dataset having twice as many columns and went unnoticed for a few days. But this was eventually fixed.

8.4 Streamlit Deployment

Deploying on streamlit is something that I have done myself and am famliar with. What I was not familiar with was the version issues with streamlit. The app ran perfectly on my local machine but did not deploy on streamlit. This was fixed by restricting the versions of the streamlit and altair packages.

9 Conclusion

This project successfully implements an end-to-end machine learning system for 72-hour AQI forecasting. By integrating automated data pipelines, robust classification models, and interpretable analysis, the system provides actionable insights for public health.