Air Quality Forecasting System

An automated MLOps pipeline predicting AQI for Karachi's Pearl regions using one year of OpenWeather data, recursive forecasting, and continuous model retraining.

System Architecture Overview

Data Collection

Hourly OpenWeather API ingestion with 40+ raw features: temperature, humidity, wind speed, pressure, PM2.5, PM10, NO_2, O_3, CO, visibility.

Feature Engineering

30+ engineered features including lag variables, rolling statistics, time-based features, and interaction terms capturing short-term and seasonal dynamics.

Model Training

Random Forest selected as best model (Test R^2: 0.70, MAE: 17.63) trained daily on one year of preprocessed data from Hopsworks.

Prediction & Visualization

72-hour recursive forecasting with Streamlit dashboard displaying color-coded AQI categories and hazard warnings for public awareness.

Data & Feature Engineering

Raw Data Sources

OpenWeather API provides 12 months of hourly observations. Standard AQI computed manually using EPA breakpoint formula since API lacks native AQI calculation.

- **Pollutants & weather:** raw input data
- **Lag & rolling features:** historical context
- **Change & std:** AQI trend and stability
- **Time encodings:** temporal patterns
- **Target:** `standard_aqi_next_24h`

Feature Categories

- Raw features: 10
- Time-based: 10
- Lag features(past AQI readings): 20
- Rolling statistics(smoothed or averaged AQI trends from recent hours): 15
- Change rates: 10
- Interactions: 5
- Targets: 3

Missing Data Handling: Forward-fill → backward-fill for gaps, median imputation for numeric columns. Leakage columns dropped to prevent data contamination.

Category	Files	Purpose
Data Collection	`fetcher.py`, `backfill.py`, `compute_aqi.py`	Get raw data from API
Feature Engineering	`transform.py`, `preprocess.py`	Create ML features
Training	`train.py`	Build prediction models
Prediction	`predict.py`	Generate 72h forecasts
Cloud Storage	`hopsworks_ingest.py`	Save to Hopsworks
Orchestration	`run_feature_pipeline.py`	Connects everything
Automation	`hourly_ingest_pip.yml`, `train_daily.yml`	CI/CD pipelines
User Interface	`app_streamlit.py`	Web dashboard

Model Performance & Selection

Model	Test R^2	MAE	RMSE
Decision Tree	0.65	18.5	22
XGBoost	0.57	21.26	26.07
LightGBM	0.59	21.01	25.32
Random Forest	0.7	17.63	21.76

Random Forest deployed: Best generalization on 1-year data. XGBoost and LightGBM overfit; SARIMAX failed at annual scale. Highest accuracy for 6h and 24h forecasts.

Recursive Forecasting Pipeline

01

Load Model & History

Retrieve trained Random Forest, scaler, feature medians, and last 48 hours of data from Hopsworks or local backup.

02

Generate Timestamps

Create 72 future hourly timestamps for prediction horizon.

03

Iterative Prediction

For each hour, engineer features from combined history, predict AQI, append prediction as input for next step.

04

Save Forecasts

Output 72 rows of predictions to predictions.csv for dashboard visualization.

Performance: Reliable up to 24h; moderate accuracy to 72h with minor drift. Recursive approach reuses predicted values as features for subsequent steps.

Automation & CI/CD Infrastructure

Hourly Ingestion

GitHub Actions runs every hour: fetches latest data, computes AQI, engineers features, updates Hopsworks Feature Store, generates predictions.

Daily Training

Runs at 02:00 UTC: retrains Random Forest on full year of data, evaluates performance, uploads model to Hopsworks if $R^2 \geq 0.70$.

Continuous Monitoring

All workflows log execution to GitHub Artifacts. Ensures uninterrupted predictions and model freshness without manual intervention.

Dashboard & Key Technologies

Streamlit Dashboard

Real-time visualization of 72-hour AQI forecasts with color-coded categories (Good to Hazardous). Displays pollutant trends, issues warnings for AQI > 200, updates automatically after each CI/CD run.

Tech Stack

- **Languages:** Python, Pandas, NumPy
- **Models:** Random Forest, XGBoost, LightGBM
- **MLOps:** Hopsworks Feature Store & Registry
- **Automation:** GitHub Actions
- **Visualization:** Streamlit, Matplotlib, Seaborn