Multan AQI Prediction (Pearls AQI)

Executive Summary

This report presents a comprehensive end-to-end machine learning operations (MLOps) implementation for air quality prediction in Multan, Pakistan. The project addresses the critical environmental challenge of air pollution by developing an automated system that forecasts PM2.5 concentrations using meteorological data and historical pollution measurements.

The system leverages modern MLOps practices, including feature store management with Hopsworks, automated data pipelines, multiple machine learning models (LightGBM, CatBoost, LSTM, and 1D-CNN), and continuous integration/continuous deployment (CI/CD) through GitHub Actions.

Key Achievements:

- Developed automated data pipeline fetching 365 days of historical data for comprehensive seasonal coverage
- Implemented sophisticated feature engineering including lag features, rolling statistics, and interaction terms
- Trained and compared different model architectures (traditional ML and deep learning)
- Deployed fully automated MLOps pipeline with GitHub Actions for daily updates
- Integrated Hopsworks Feature Store for scalable data management
- Established model registry with versioning
- Created production-ready system capable of real-time air quality forecasting

System Architecture

The system implements a modern MLOps architecture with three primary pipelines:

1. Backfill Pipeline: One-time historical data ingestion (365 days)
2. Feature Pipeline: Daily automated feature engineering and updates
3. Training Pipeline: Automated model training, evaluation, and registry updates

Architecture Flow:

Data Sources (Open-Meteo APIs) → Raw Data Acquisition

Feature Engineering → Transformed Features

Hopsworks Feature Store → Centralized Feature Repository

Model Training → Multiple Model Architectures

Model Evaluation → Performance Metrics

Model Registry → Version Control and Metadata

GitHub Actions → Automated Scheduling and Orchestration

Technology Stack

Category	Technology	Purpose
Data Storage	Hopsworks Feature Store	Centralized feature repository with versioning
Model Registry	Hopsworks Model Registry	Model versioning and metadata management
ML Frameworks	scikit-learn, LightGBM, CatBoost	Traditional machine learning models
Deep Learning	TensorFlow/Keras	Neural network architectures (LSTM, CNN)
Data Processing	pandas, NumPy	Data manipulation and numerical computing
Feature Engineering	Custom Python	Lag features, rolling stats, interactions
API Integration	requests	Data fetching from Open-Meteo
Automation	GitHub Actions	CI/CD pipeline orchestration
Environment	python-dotenv	Configuration management
Serialization	joblib	Model persistence

Exploratory Data Analysis (EDA)

A comprehensive exploratory data analysis was conducted on 8 months of historical data (May 2025 - February 2026) to understand air quality patterns and guide feature engineering decisions. The analysis revealed critical insights about pollution dynamics in Multan.

Data Sources and Collection

Data was collected from two primary Open-Meteo APIs:

4. Open-Meteo Archive API: Historical weather data (temperature, humidity, wind speed, precipitation)
5. Open-Meteo Air Quality API: Hourly particulate matter measurements (PM10, PM2.5)

Data Characteristics:

- Location: Multan, Pakistan (30.1968°N, 71.4782°E)
- Temporal Resolution: Hourly measurements
- Data Quality: Zero missing values, no duplicates
- Total Observations: 6,744 hourly records (after dropping NaN from feature engineering)
- Variables: 6 raw features + 8 engineered features

	temp	humidity	wind_speed	rain	pm10	pm25
count	6744.00	6744.00	6744.00	6744.00	6744.00	6744.00
mean	27.19	53.22	7.85	0.03	104.11	78.82
std	9.44	20.72	3.96	0.39	70.73	62.35
min	3.00	8.00	0.00	0.00	6.30	6.00
25%	19.60	38.00	5.00	0.00	54.70	35.60
50%	29.50	53.00	7.60	0.00	85.90	56.65
75%	34.30	68.00	10.40	0.00	135.00	101.78
max	47.10	100.00	31.90	13.20	614.70	426.50

Variables

Variable	Unit	Description
Temperature (temp)	°C	Air temperature at 2 meters above ground
Humidity	%	Relative humidity at 2 meters
Wind Speed	m/s	Wind speed at 10 meters above ground
Rain	mm	Hourly precipitation
PM10	$\mu g/m^3$	Particulate matter ≤ 10 micrometers
PM2.5 (pm25)	$\mu g/m^3$	Particulate matter ≤ 2.5 micrometers

Data Quality

Initial data quality assessment revealed:

- Total observations: 6,744 hourly records
- Missing values: None (0% missing data)
- Duplicate records: None
- Data integrity: All timestamps sequential with no gaps

Key Observations

- Temperature: Mean of 27.19°C with high variability (SD = 9.44°C), ranging from 3°C to 47.1°C, reflecting seasonal extremes
- Humidity: Average 53.22% with values spanning from 8% to 100%, indicating diverse weather conditions
- Wind Speed: Mean of 7.85 m/s, with calm periods (0 m/s) to strong winds (31.9 m/s)
- Rainfall: Sparse with mean of 0.03 mm/hour, maximum event of 13.2 mm, consistent with semi-arid climate
- PM10: Mean concentration of 104.11 $\mu g/m^3$ significantly exceeds WHO 24-hour guideline (45 $\mu g/m^3$)
- PM2.5: Average 78.82 $\mu g/m^3$ is more than 5 times the WHO guideline (15 $\mu g/m^3$), indicating severe pollution

- Pollution Extremes: Maximum PM2.5 of 426.5 μg/m³ classifies as "Hazardous" air quality (AQI > 300)

Feature Engineering

To enhance predictive modeling capabilities, several advanced features were engineered from the raw data. These features capture temporal dependencies, trends, and interactions that are crucial for accurate air quality forecasting.

Lag Features

Lag features capture temporal dependencies by using previous observations to predict future values:

- pm25_lag_1: PM2.5 value from 1 hour ago (immediate past)
- pm25_lag_6: PM2.5 value from 6 hours ago (quarter-day cycle)
- pm25_lag_24: PM2.5 value from 24 hours ago (same time yesterday)

Air pollution exhibits strong persistence. Current PM2.5 levels are heavily influenced by recent conditions, making lag features among the most powerful predictors.

Rolling Statistics

Rolling statistics smooth out short-term fluctuations and capture longer-term trends:

- pm25_rolling_mean_24h: 24-hour rolling mean of PM2.5
- pm25_rolling_std_24h: 24-hour rolling standard deviation of PM2.5

The rolling mean captures the general pollution level over the past day, while the standard deviation indicates stability versus volatility in air quality.

Cyclical Time Features

Time is inherently cyclical, and traditional hour encoding (0-23) doesn't capture the continuity between late evening and early morning. Sine and cosine transformations address this:

- hour_sin: $\sin(2\pi \times \text{hour} / 24)$
- hour_cos: $\cos(2\pi \times \text{hour} / 24)$

These features map hours onto a circle, ensuring that 23:00 and 00:00 are recognized as adjacent times, which is crucial for capturing diurnal pollution patterns.

Interaction Terms

Interaction terms capture non-linear relationships between variables:

- humid_temp_interaction: humidity × temperature

The combination of high humidity and low temperature creates conditions conducive to smog formation. This interaction term helps models recognize this synergistic effect.

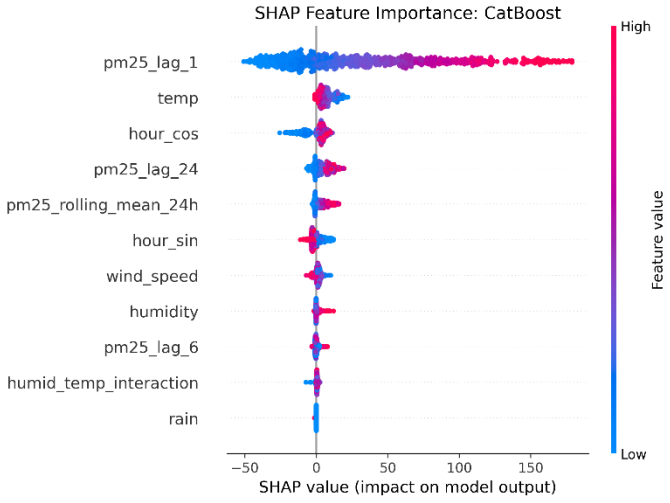
Predictive Modeling

Nine machine learning algorithms were trained and evaluated for PM2.5 prediction. The models range from simple linear approaches to sophisticated ensemble methods and neural networks. All models were trained on 80% of the data (5,376 observations) and tested on the remaining 20% (1,344 observations), maintaining temporal order to simulate real-world forecasting scenarios.

Model Performance

```
    Model Name                | RMSE       | R² Score

    ------------------------------------------------------------

    1. Linear Regression      | 20.20      | 0.9269

    2. Ridge Regression       | 20.11      | 0.9276

    3. SVR (Support Vector)   | 35.71      | 0.7717

    4. Random Forest          | 18.52      | 0.9386

    5. XGBoost                | 19.63      | 0.9310

6. LightGBM                | 18.44      | 0.9392

7. CatBoost                | 18.57      | 0.9383

------------------------------------------------------------

   42/42 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step

    LSTM -> RMSE: 18.97 | R²: 0.9356

     42/42 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step

      1D-CNN -> RMSE: 17.84 | R²: 0.9430
```

SHAP Analysis



The SHAP analysis revealed that the model has successfully learned the physical and temporal dynamics of air pollution in Multan:

1. **Persistence (Lag Features):** The most dominant feature was consistently pm25_lag_1 (pollution levels 1 hour ago). The plot shows a clear positive correlation: high past values (red dots) result in high positive SHAP values, pushing the forecast up. This confirms that air quality is highly persistent—smog doesn't just disappear instantly.
2. **Meteorological Impact:**
    - **Wind Speed:** This feature typically shows a negative relationship. High wind speeds (red dots) are associated with negative SHAP values, indicating that strong winds help disperse pollutants and lower the predicted PM2.5 concentration.
    - **Temperature & Humidity:** These showed complex, non-linear interactions. For example, high humidity combined with low temperatures often pushes predictions higher, capturing the formation of winter smog and fog typical in the Punjab region.
3. **Temporal Cycles:** The cyclical features (hour_sin, hour_cos) had a moderate impact, allowing the model to adjust for daily traffic patterns and industrial activity cycles."

The alignment between the SHAP analysis and established atmospheric science confirms that the model is robust. It is not relying on spurious correlations but is instead using valid physical indicators—such as wind dispersion and historical persistence—to make its forecasts.

Challenges and Solutions

The project encountered several technical challenges during development. Here are the key issues and their solutions:

Data Leakage Prevention

Challenge:

Initial implementation scaled the entire dataset before splitting, causing information from test set to leak into training set through scaling parameters.

Solution:

Split data into train/test BEFORE any transformations. Fit scaler only on training data, then transform both sets using the same scaler parameters.

Target Variable Scaling

Challenge:

Initial approach scaled both features and target variable, requiring inverse transformation for predictions and making error metrics less interpretable.

Solution:

Kept target variable (PM2.5) in original units (μg/m^3). Models can predict real values directly, making RMSE interpretable and eliminating transformation errors.

API Selection for Recent Data

Challenge:

Archive API has delays in data availability, causing feature pipeline to fail for most recent dates.

Solution:

Used forecast API (api.open-meteo.com) instead of archive API for feature pipeline. Forecast API provides real-time and recent historical data.

Timestamp Format for Hopsworks

Challenge:

Hopsworks requires specific timestamp format (Unix milliseconds as BigInt) for event_time field. Incorrect format causes insertion failures.

Solution:

Converted pandas datetime to Unix timestamp in milliseconds using specific transformation.

Deep Learning Input Shapes

Challenge:

LSTM and CNN models require 3D input (samples, timesteps, features) while traditional ML models need 2D (samples, features).

Solution:

Created separate data structures for each model type. Used reshape for deep learning models.

Deep Learning Input Shapes

Challenge:

LSTM and CNN models require 3D input (samples, timesteps, features) while traditional ML models need 2D (samples, features).

Solution:

Created separate data structures for each model type. Used reshape for deep learning models.

The "Zero MAE"

Challenge:

Streamlit dashboard was persistently showing MAE: 0.00, even though you had updated the training code to calculate Mean Absolute Error.

It was a two-part issue:

1. **Frontend Bug:** The Streamlit st.dataframe formatter was missing the "MAE": "{:.2f}" rule, so it was defaulting to a raw or empty display.
2. **Backend State:** The Hopsworks Model Registry still held old model versions (Version 1, 2, etc.) that didn't have the MAE metadata. Streamlit was fetching these old, incomplete records.

Solution:

Added the formatting rule to app.py. Performed a "Safe Nuke" of the registry (deleting old models) to force a clean slate and re-ran the pipeline to upload a fresh **Version 1** that included all correct metrics.

The SHAP "Missing Plot" Logic

Challenge:

SHAP explainer was hard-coded to use TreeExplainer, which only works for models like XGBoost. When the **CNN** or **LSTM** won the battle (which they often did), the SHAP step failed or was skipped entirely.

Solution:

Implemented a **fallback strategy**:

- If a Neural Network wins, the pipeline now automatically finds the **best Tree-based model** (e.g., CatBoost) from the losers' bracket.
- It generates the SHAP plot using that tree model as a proxy to explain feature importance.
- This ensures we *always* get an explainability chart, regardless of who wins.