AQI Forecasting System Report

This report summarizes the design, implementation, and performance of a fully automated MLOps pipeline for hourly Air Quality Index (AQI) forecasting in Karachi, Pakistan. The system delivers a **96-hour forecast** every hour and visualizes historic and future data on a live Streamlit dashboard.

Feature Development & Data Engineering

This part handles data acquisition, cleaning, feature engineering, and exploratory analysis:

- **Data Acquisition & US-EPA Conversion:** Hourly pollutant data ($PM_{2.5}$, PM_{10}, O_3, NO_2, SO_2, CO) is fetched from **OpenWeatherMap**. Proprietary AQI is converted to US-EPA AQI using EPA breakpoints; overall AQI is the maximum of sub-AQIs.
- **Data Cleaning:** Duplicates removed, negative values clipped, outliers capped at 99.5th percentile, and missing values imputed via forward/backward/median fill.
- **Feature Engineering:** Temporal/cyclical features (hour, month, day, weekend + sine/cosine transforms), interaction/derived features (pm_ratio, total_pm, total_gases, no2_o3_ratio), and time-series features (3-hour rolling means, lagged AQI values) for historical context.
- **Exploratory Data Analysis (EDA):** Statistical summaries, correlation analysis, and visualizations (distributions, trends, scatter plots, heatmaps) were used to guide feature selection for model training.

Feature Store & Data Engineering (Hopsworks)

This part includes manages clean feature preparation, validation, and integration with Hopsworks Feature Store:

- **Feature Selection:** 17 features selected from prior EDA, including pollutant concentrations, AQI lag values, cyclical temporal features, and interaction/rolling metrics.
- **Data Cleaning & Validation:** Datetime parsing, missing values filled, invalid timestamps dropped, numeric columns cast to float64, NaN/inf replaced with 0. Deduplication ensures unique timestamps.
- **Hopsworks Integration:** Cleaned data uploaded as a versioned feature group (aqi_features v4), including an auto-generated id primary key. Feature group can be listed, deleted, and re-uploaded safely.
- **Data Fetch & Verification:** Features fetched for training or analysis, with optional date filtering. Local CSV and Hopsworks data compared to ensure consistency (max numeric difference < tolerance).
- **Training Data Ready:** Validated dataset is fully prepared for downstream AQI model training.

Model Training, Evaluation, and Checkpointing

A suite of seven regression models was trained to identify the optimal predictor for the time-series forecasting task.

- **Model Suite:** XGBoost, LightGBM, CatBoost, RandomForest, GradientBoosting, Ridge, and Linear Regression.
- **Time-Series Split:** The data was split **chronologically** (70% Train, 10% Validation, 20% Test) to accurately simulate a real-world deployment scenario and prevent **data leakage**.
- **Hyperparameter Tuning & CV:** Models were tuned using optimized parameters (e.g., restricted max_depth, subsample, colsample, and regularization for tree models) and evaluated using **TimeSeriesSplit (n_splits=5)** cross-validation on the training data. The model with the **lowest CV-MAE** was chosen as the best_model.

A. Best Model Selection and Checkpointing

The training run evaluated seven regression models using a **TimeSeriesSplit Cross-Validation** (CV), with **Mean Absolute Error (MAE)** as the decisive metric.

Model	**CV-MAE (Checkpoint Metric)**	**Train MAE**	**Test MAE**	**Test R2**
XGBoost	**2.060**	1.121	3.446	0.975
LightGBM	3.146	0.966	3.646	0.975
Random Forest	2.251	1.162	5.632	0.952

Model Selection: The **XGBoost** model was identified as the best performer for the current run, achieving the lowest CV-MAE of **2.060**. This model's predictive power is further demonstrated by a strong Test R2 of 0.975.

Checkpoint Status: The current best model (XGBoost CV-MAE=2.060) was compared against the **Historical Best Checkpoint** (XGBoost CV-MAE=2.054). Because the current run was **not strictly better** (2.060 not equal to 2.054), the deployment pipeline maintained the existing best_model.pkl checkpoint. This strict **deployment gate** ensures that the production model is only replaced when a statistically superior version is confirmed, guaranteeing continuous quality improvement.

B. High Accuracy and Overfitting Mitigation

The models exhibit extremely high **Train R2** values (e.g., XGBoost at 1.000). This high degree of fit on the training data is **not considered overfitting** due to the architectural safeguards in place:

1. **Cross-Validation (CV) Reliance:** The system **never** uses the high training accuracy for deployment decisions. The **MAE is based on the CV set**, which assesses the model's performance on multiple historical subsets of data it has never seen. The low (CV-MAE = 2.060) confirms the model's ability to **generalize** effectively.

2. **Validation/Test Metrics:** The performance on the dedicated, chronologically separated Test Set (MAE=3.446, R2=0.975) remains high, confirming that the model has learned generalized patterns, not just noise specific to the training period.

C. Continuous Data Shift Management

Environmental time-series data is highly susceptible to **data shift** (a change in the underlying data patterns due to unmodeled external factors like weather anomalies or policy changes). The system manages this risk through:

1. **Daily Retraining:** The entire training process executes **daily** via GitHub Actions. This constant exposure to the latest data distribution allows the model to quickly adapt its weights, preventing the significant performance decay that is typical when an older model encounters shifted data.
2. **Short Forecast Horizon:** The 96-hour (4-day) forecast horizon limits the forecast window, focusing the model's prediction on the short-term future where the assumption of a stable data distribution is most robust.

Automated CI/CD Pipeline (GitHub Actions)

The MLOps workflow is fully automated via three chained GitHub Actions:

1. **fetch_data.yml:** Hourly/manual trigger; runs fetch_data.py, pulls data, applies US-EPA conversion, commits data, and dispatches data-updated event.
2. **eda.yml:** Triggered by data-updated or manually; runs eda.py for cleaning, feature engineering, selection, uploads features to Hopsworks Feature Store, commits artifacts, and dispatches eda-completed event.
3. **train.yml:** Hourly/manual trigger; runs training.py to retrain models, checkpoint the best model, generate 96-hour forecasts, commit models, metrics, forecasts, comparison plots, and upload artifacts.

Web Application Dashboard (Streamlit Cloud) AQI Prediction Dashboard · Streamlit

A dynamic Streamlit dashboard provides transparent access to system outputs.

- **Deployment:** app.py is deployed on Streamlit Cloud, linked to GitHub for automatic containerization and hourly updates via CI/CD commits.
- **Key Features:**
  - **Future Predictions:** Interactive selection of models (default: best checkpoint) with 96-hour Actual vs. Predicted AQI line charts.
  - **Historical Data:** Zoomable 30-day time-series plots for AQI and pollutants for EDA.
  - **Forecast Comparison:** Bar charts of model metrics (MAE, R^2) for evaluating model drift and performance shifts.