Project Report: Air Quality Predictor

Author: Muhammad Zain

Role: Data Science Intern

Project: Rawalpindi AQI Forecasting System

1. Project Overview

The "aqi-predictor" is an automated Machine Learning pipeline designed to predict the Air Quality Index (AQI) for Rawalpindi, Pakistan. It leverages a serverless architecture to handle the entire ML lifecycle from data collection to real-time inference ensuring the model stays updated with the latest environmental data without manual oversight.

2. Problem Statement

Rawalpindi faces significant seasonal air quality challenges. Traditional static models become obsolete as weather patterns change. This project solves that by implementing **Continuous Training (CT)** and **Continuous Deployment (CD)**, providing residents with an accurate 72-hour AQI forecast.

3. Technical Architecture

The system is divided into three decoupled pipelines:

- **Feature Pipeline (`1_feature_pipeline.py`):** Fetches real-time weather and pollution data (PM2.5, NO_2, etc.) from the Open-Meteo API and stores it in the **Hopsworks Feature Store**.
- **Training Pipeline (`2_training_pipeline.py`):** Runs daily via **GitHub Actions**. It evaluates multiple models (Random Forest, Gradient Boosting, Ridge) and registers the "Champion" model (currently Random Forest with ~80% accuracy) to the **Hopsworks Model Registry**.
- **Inference Pipeline (`3_app.py`):** A **Streamlit** dashboard that pulls the latest model and generates a 3-day forecast.

4. Project Deliverables

Following the requirements for a 100% serverless MLOps stack, the following deliverables have been implemented and integrated into the repository:

A. Feature Pipeline & Data Management

- **Automated Feature Pipeline:** A robust Python script (`1_feature_pipeline.py`) that fetches real-time pollutants (PM2.5, PM10, NO_2, O_3) and weather variables from the Open-Meteo/AQICN APIs.
- **Feature Engineering Engine:** Implementation of time-based features (hour, day, month) and derived environmental metrics to improve model sensitivity to local patterns.
- **Hopsworks Feature Store Integration:** A centralized "Single Source of Truth" where processed features are stored, versioned, and made available for both training and inference.
- **Historical Backfill Script:** A data ingestion module used to populate the Feature Store with historical records, enabling the creation of a comprehensive training dataset.

B. Training Pipeline & Model Registry

- **Continuous Training (CT) Pipeline:** A daily automated workflow (`2_training_pipeline.py`) that fetches historical features from Hopsworks to retrain the system.
- **Multi-Model Experimentation:** A comparison framework evaluating **Random Forest**, **Ridge Regression**, and **Gradient Boosting** models using RMSE, MAE, and R^2 metrics.
- **Model Registry:** Automated registration of the "Champion" model into the Hopsworks Model Registry, ensuring version control and easy rollbacks.

C. Automated CI/CD & Orchestration

- **Serverless Orchestration:** Deployment of **GitHub Actions** workflows to schedule the Feature Pipeline (hourly) and Training Pipeline (daily), achieving a zero-ops environment.
- **Environment Management:** Secure handling of API keys and Hopsworks credentials using GitHub Secrets and `.env` configurations.

D. Web Application & Analytics

- **Streamlit Interactive Dashboard:** A production-ready web interface (`3_app.py`) that loads the latest model from the registry to provide a 3-day (72-hour) AQI forecast.
- **Real-time Inference Engine:** A backend logic that fetches live forecast weather data and applies the Champion model to generate immediate predictions.
- **Exploratory Data Analysis (EDA):** Visualizations within the dashboard identifying air quality trends and seasonal fluctuations in Rawalpindi.

- **Feature Importance:** Insights into which environmental factors (e.g., humidity vs. specific pollutants) most significantly impact the AQI prediction.

5. Performance & Results

- **Primary Metric:** R^2 Score (Accuracy).
- **Champion Model:** Random Forest Regressor.
- **Current Accuracy:** ~78% - 80%.
- **Automation:** 100% serverless execution using GitHub Actions schedules.

6. Challenges & Learnings

- **Data Drift:** Managed by daily retraining to adapt to sudden environmental changes.
- **System Integration:** Successfully integrated Hopsworks with GitHub Actions to maintain a free, serverless infrastructure.
- **API Management:** Handled rate limits and data cleaning for real-time weather APIs.

7. Conclusion

The **Rawalpindi AQI Predictor** successfully demonstrates the power of a serverless MLOps framework in solving real-world environmental challenges. By automating the transition from data ingestion to model deployment, the project eliminates the "technical debt" typically associated with manual ML workflows.

This system provides a scalable foundation that can be easily adapted for other cities or different environmental parameters. Ultimately, this project serves as a bridge between raw data and actionable insights, empowering the citizens of Rawalpindi with the information needed to make health-conscious decisions in the face of rising urban pollution.