# PROJECT REPORT: AQI FORECASTING SYSTEM

## 1.OVERVIEW

This project develops an automated, end to end Machine learning pipeline to monitor and predict the Air Quality Index(AQI) for Karachi, Pakistan.This system perform live data collection, preprocessing, feature engineering, feature store (Hopsworks), automated retraining pipelines (Github actions) and visualize via streamlit dashboard.

**Historical And live AQI DATA:** Air Quality indices and pollutants concentrations(PM2.5, PM10, NO2, etc.) are retrieved from OpenWeatherMap Air pollution Api, providing a real time updates.

**Mateorological Data:** Weather variables including temperature, windspeed etc are extracted from Open Mateo Historical Archive.

## 2. DATA COLLECTION

**fetch.py**

- In this script two Public APIs are used together to fetch historical and weather data from 180 days hourly.
- The pollution and weather data were merged together to match timestamps so that each record contains pollutions level along with environmental conditions.
- For every timestamp all features values are store in list of dictionaries called rows and these rows are converted into pandas Dataframe and saved as: **"data/first_aqi_data.csv"**

## 3.EXPLORATORY DATA ANALYSIS(EDA) AND PREPROCESSING

**eda_preprocessing.ipynb**

- The dataset contains 4315 hourly records with 12 features with no missing value found in any column.
- The AQI distribution has follow discrete distribution with most values around AQI LEVEL 3.
- **PM2.5,PM10 and CO** were highly right skewed indicate the presence of extreme pollution events.
- Boxplot and histogram shows that:**PM2.5,PM10** contained the highest no of extreme values, **No2 and CO** shows moderate outliers and AQI and **O3** had almost no outliers.
- Outlier Handling: An IQR based capping technique is used to limit extreme values, which affects the small portion of data (**PM2.5: 4.22%, PM10: 4.06%, NO2: 3.31%, CO: 2.53%)** which reduced noise while preserving the full dataset.
- Strong positive correlations were identified between AQI and major pollutants such as PM10 (0.90), PM2.5 (0.87), CO (0.75), and O3 (0.72),while negative relationship showed between weather variable(temperature, windspeed) and AQI.

- After cleaning and capping dataset saved into "**data/preprocessed_aqi_data.csv**"

## 4. FEATURE ENGINEERING

**feature_engineering.ipynb**

- TIME BASED FEATURES (**year ,month, day, hour, day_of_week, minute**): These features capture hourly,weekly,seasonality and daily trends ,helping the models to understand how AQI changes over time.
- DERIVED FEATURES (**aqi_change,aqi_pct_change**):These features help the model to understand sudden spikes,momentum and shift the model from knowing what pollution is to understand how it is changing.
- WEATHER INTERACTION FEATURE**(temp humid ,wind pollution):**These features captures the impact of weather condition and wind speed, enabling the model to learn how environmental factors influence AQI variation.
- LAG FEATURES(**aqi_lag_1&aqi_lag_3)**:Lag features allow the model to use past AQI values to improve the prediction of future AQI .
- ROLLING WINDOW FEATURES(**rolling_mean and max):** These values were computed for major pollutants **(PM2.5, PM10, NO2, CO, O3, SO2)** to capture their short-term behavior and impact on AQI prediction.
- TARGET VARIABLE(**target_Aqi_24h):**This feature represent the AQI value 24h ahead and is used as forecasting target.
- Before storing the features in hopsworks unnecessary lag and rolling feature were removed to avoid data leakage ,keeping only the most relevant features and saved this clean dataset into final csv file:"**data/karachi_aqi_feature.csv"**

## 5. FEATURE STORE IMPLEMENTATION

**hopsworks_script.py**

- The script connects the Hopsworks Feature Store using project name and API KEY.
- Karachi_aqi_features_v7 is created because earlier versions had schema changes and deleted features.
- The processed feature dataset is loaded from csv file for storage in Hopsworks.
- Cleaned and formatted 24 features are inserted into Feature Store with upsert functionality.

## 6.MODEL TRAINING & EVALUATION

**train.py**

- Data was directly fetched from Hopsworks Feature Store.
- Leakage prone features such as (aqi_change, aqi_pct_change, target_aqi_24h)were removed before model training.
- The data was using time based split with 80% for training and 20% for testing.

- Features were normalized using MinMaxScaler.
- **MODELS:** Ridge Regression, Random Forest, XGBoost, LightGBM and LSTM. Model performance was evaluated using MAE, RMSE, and $R^2$ metrics.

| MODELS | RMSE | MAE | $R^2$ | Behavior |
|---|---|---|---|---|
| Ridge Regression | 0.328 | 0.249 | 0.824 | Underfit |
| Random Forest | 0.103 | 0.025 | 0.983 | Best Fit |
| XGBoost | 0.108 | 0.040 | 0.981 | Good Fit |
| LightGBM | 0.135 | 0.056 | 0.970 | Good Fit |
| LSTM | 0.930 | 0.056 | 0.970 | Underfit |

- **Random Forest** was selected as the final best model for deployment.

## 7. SHAP ANALYSIS

**shap.ipynb**

- SHAP analysis was performed to interpret trained model and understand how each feature contribute to AQI prediction.
- A TreeExplainer was used for fast and accurate explanations.
- A SHAP summary plot was created to show how each feature affect model's prediction.
- The bar plot of SHAP values identify the most important feature : **aqi_lag_1, pm10, pm2_5, and ozone.**
- The result confirmed that past AQI and key pollutants are strongest factor for predicting future AQI.

## 8. CI/CD AUTOMATION

| PIPELINE | SCRIPT | GITHUB ACTION WORKFLOW | TIME(PKT) | MAIN TASK | OUTPUT |
|---|---|---|---|---|---|
| Daily feature Pipeline | feature_pipeline.py | daily_retrain.yml | Every hour | Fetch latest AQI data → preprocessing → feature engineering → insert into Hopsworks Feature Store | Update feature group |
| Daily Model Training | train_model.py | daily_model_training.yml | Daily at (9:05a.m) | Load features → train models → evaluate → save best model | Update model in model registry |

## 9. STREAMLIT DASHBOARD

**app.py**

- The streamlit dashboard provides an interactive interface for live AQI monitoring and forecasting for Karachi.
- It fetched live data from hopsworks Feature Store and loads the latest trained model and In sidebar their performance metrics are evaluate prediction accuracy.
- A 72 hour AQI forecast visualizes predicted air quality trend and next 3 days forecast summary present average AQI values**.**
- Pollutant breakdown chart show contribution of **PM2.5,PM10,C0,NO2,SO2,03** to overall AQI.
- A 30 day AQI overview using a pie chart to summarize historical AQI distribution.

## 10. PROBLEMS AND FIXES

**1.Hopsworks Schema & Versioning Issue**

**Problems:**

- Frequent feature changes cause schema errors while inserting data into Hopsworks.
- Older feature groups version contained unused or wrong typed columns.
- Duplicate entry error occurred during data insertion.

**Fix:**

- Created new feature group version 7 to store only final features.
- Set DateTime as the Primary Key to prevent duplicate records.
- Discarded previous version to avoid schema mismatch problems.

**2.Hopsworks Client Compatibility Issue**

**Problems:**

- Client-backend version mismatch warning(Hopsworks 4.6.x vs backend 4.2.x).
- Python 3.12 cause dependency errors, including Kafka connection issues.

**Fix:**

- Created a new virtual env with **python 3.10**(supported by Hopsworks 4.2.)
- Reinstalling all other libraries and added **confluent-kafka** to resolve dependency errors.

**3.Github Action Cron Time Mismatch**

**Problem:**

- Scheduled github actions were not running at expected local times due to UTC time difference.

**Fix:**

- Cron schedules were adjusted according to UTC timezone.

**4.Feature Pipeline Data Insertion Issue**

**Problem:**

- Pipeline run successfully, but no data is added into Hopsworks.

**Fix:**

- Added options :write_options={"wait_for_job":True} in insert option.
- Check Github Actions logs for error and confirmed data was added into Hopsworks.

**5.Model Registry Schema Error**

**Problem:**

- The model registry reject the model because of data type mismatches and incorrect feature alignment.

**Fix:**

- An input example was passed using **input_example = X_train.sample(5)** while saving the model.
- This line provide a small sample of dataset to the registry so that Hopsworks can automatically detect the correct schema.

**6.Local Vs Cloud Environment Mismatch**

**Problem:**

- Model worked locally but failed in streamlit cloud due to different library version.

**Fix:**

- Match Library version across training and deployment environment.

## 11. CONCLUSION

The AQI Forecasting Dashboard implemented end-to-end MLOps pipeline that:

- Collect real time AQI and weather data from API, perform EDA and cleans the data.
- Generate meaningful features through feature engineering and stores them into Hopsworks.
- Uses Github Actions to automate the workflow, where feature pipeline runs every hour to ingest fresh data and model pipeline retrains daily.

- Deploy trained model and present real time predictions through an interactive dashboard.