Air Quality Prediction Report

Introduction

This report outlines the approach and methodology used for building a predictive model for Air Quality Index (AQI). The project involved collecting, processing, and analyzing air quality data to develop a machine learning model capable of predicting AQI categories. Below is a detailed breakdown of the steps taken, the tools utilized, and the outcomes achieved, along with areas where improvements are needed.

Data Collection and Preparation

Data Source

The data for this project was collected using the OpenWeather website's Air Quality API. The API was utilized to gather real-time air quality data for a period of two months, spanning from **November to January**.

Data Extraction and Formatting

- The API provided data in **JSON format**, which was extracted and processed into a structured format.
- A **Pandas DataFrame** was created from the JSON data to facilitate analysis and visualization.
- The extracted data included key air quality indicators such as CO,NO, NH3 NO2, O3, PM2.5, PM10, and SO2.

Visualization

The cleaned data was used to create visualizations that highlighted trends and patterns in air quality over the two-month period. These visualizations provided insights into pollutant levels and their variations over time.

Model Selection and Training

Machine Learning Models

Three different machine learning models were evaluated to predict AQI:

1. **Random Forest Regressor**
2. **XGBoost Regressor**
3. **Support Vector Regressor (SVR)**

Model Performance

- After extensive testing and evaluation, the **Random Forest Regressor** delivered the best accuracy among the three models.
- The model's ability to handle non-linear relationships and its robustness to overfitting made it the ideal choice for this dataset.

Limitations

- Hyperparameter tuning for the models was conducted manually, which limited optimization potential.

Deployment Using Streamlit

Streamlit Web Application

A **Streamlit web application** was developed to provide an interactive interface for users to:

- Input pollutant levels and other parameters.
- Predict the AQI category.
- Visualize pollutant levels in real-time.

Features of the Application

- User-friendly sliders and input boxes for parameter selection.
- Dynamic predictions displayed based on the trained model.
- Basic styling was added to enhance the user experience.

Challenges and Future Work

Challenges Faced

1. **Continuous Data Collection**:

    - While the API was manually queried to collect data for two months, automating the process for hourly data collection was not implemented.
2. **CI/CD Pipeline**:
    - A robust **CI/CD pipeline** for automating the data collection and model retraining process was not established.
3. **Feature Stores**:
    - The project did not integrate with a **feature store** (e.g., Hopsworks) to manage and serve features for the model in a production environment.

Areas for Improvement

1. Automating data collection using a scheduled process (e.g., **cron jobs** or **cloud-based triggers**) to ensure continuous data availability.
2. Implementing a CI/CD pipeline to:
    - Automate model retraining.
    - Deploy updated models seamlessly.
3. Exploring the use of advanced tools for feature management, such as **Hopsworks Feature Store**.
4. Enhancing the model by incorporating additional data sources and advanced hyperparameter tuning techniques.

Conclusion

The project successfully demonstrated the capability to predict AQI using machine learning models and deployed an interactive web application for user interaction. However, there are several aspects, such as automation and production-level feature management, that require attention for future iterations. Despite these challenges, the work provides a strong foundation for further development in air quality prediction systems.
