Air Quality Prediction using Apache Kafka

Introduction

Air quality significantly impacts public health, urban planning, and environmental sustainability. In this project, we leverage Apache Kafka and machine learning models to simulate real-time data streaming for predicting Carbon Monoxide (CO) concentrations, a key air pollutant, based on historical and temporal data.

The primary goals of this project are:

- To stream air quality data using Kafka.
- To perform data exploration and cleaning on the data.
- To visualize and analyze the relations between the variables or features.
- To build machine learning models that predict the decided pollutant levels.

Kafka Setup Description

Kafka Architecture:
Apache Kafka is set up as the backbone for streaming. It includes:
Producer: A Python script that sends data to Kafka topics. The data is either simulated from the dataset or directly fetched from the API. In our use case, it is using 20% data from the dataset as the testing data.

Topic: A single topic air_quality_data is used to hold the streaming records.

Consumer: A separate Python consumer subscribes to the topic and runs the machine learning model, and stores data in a CSV file after a separate condition is met.

Installation:
Kafka was downloaded from the official website and extracted.
Producer and Consumer were developed using kafka-python.

Data Flow Pipeline:
[Dataset] -> [Producer] -> [Kafka Topic] -> [Consumer] -> [Model Prediction]

Data Exploration and Findings

<u>Data Preprocessing</u>

The dataset had missing values, which were represented by -200 as a placeholder. Additionally, some columns had gaps that required filling to avoid errors.

Steps:

1. Replaced -200 with NaN across the dataset. This ensures that these values are recognised as missing data, which is essential to handling these missing values.

2. Handled missing data with interpolation

- For Environmental Columns: [T, RH, AH]
  I applied linear interpolation as it would estimate missing values based on surrounding data points. I used this because these variables are in continuous trend, and this interpolation technique was the best fit, according to me.
- For Sensor Columns
  I checked the proportion of missing values. If the missing values were less than 5%, I used linear interpolation again for the column. But if the missing data was more than 5%, I filled the missing values with the mean of the column. This is because larger gaps in the data are less predictable and filling with the mean avoids introducing bias from interpolation.

3. Dropped the NMHC column as it was having 90% NULL Values

```
NMHC(GT)            8443   90.231912
```

4. Combining Date and Time into a Single Column
   The dataset had separate date and time columns; thus, for the ease of working in time-based analysis, I changed it to a unified date and time column.

5. Ensuring Consistent Data Types
   The new date and time column was converted to datetime format and all the other columns were converted to int or float.

Data Analysis

The analysis provides trends and relationships in the air quality dataset. We focus on three pollutants- Carbon Monoxide (CO), Nitrogen Oxides (NOx) and Benzene (C6H6).

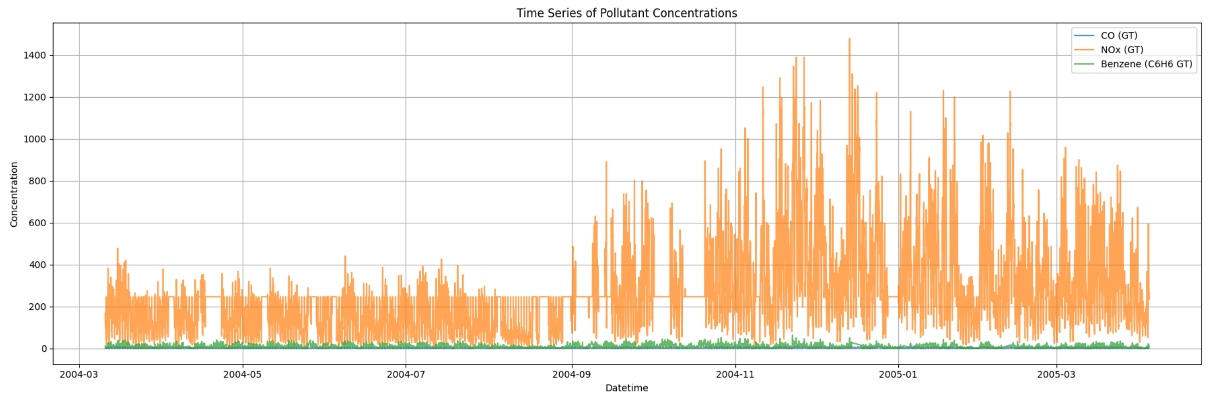
Key Pollutants Tracked:

- CO(GT): Carbon monoxide
- C6H6(GT): Benzene
- NOx(GT): Nitrogen oxides

Basic Visualizations

1. Time-Series Analysis of Pollutant Concentrations

    - The time series plot of the pollutants show clear fluctuations in their concentrations overtime, with noticeable peaks during the day, especially during early mornings and evenings. This suggests that CO levels might be because of human intervention, such as traffic patterns or human activities such as working industries.
    - NOx concentrations also show a similar pattern, mixing with CO analysis, which might be because of vehicular emissions during peak hours.
    - Benzene does show some peaks, but is typically related to chemical processes and industrial processes.

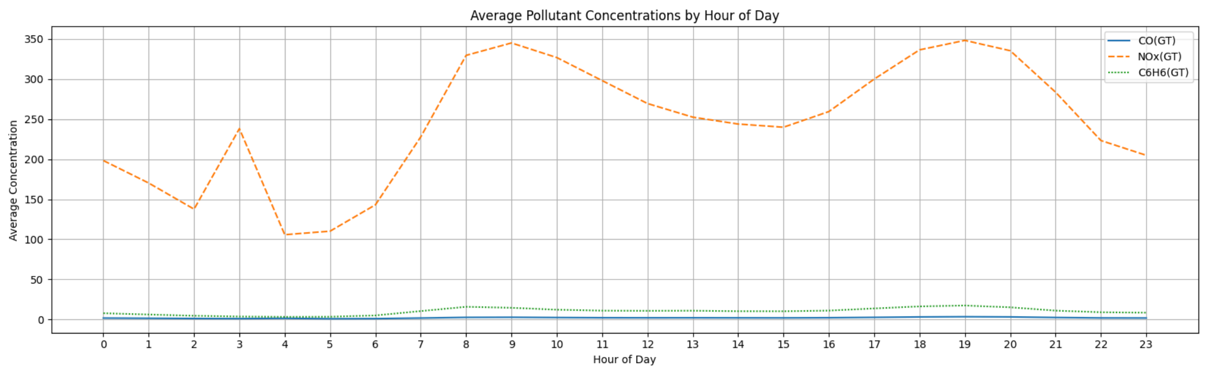


2. Hourly and Daily Patterns

Hourly trends:

- NOx peaks during morning and evening rush hours, aligning with traffic patterns.
- CO shows the same pattern but with less intensity.
- Benzene remains the same/stable with minor variations.

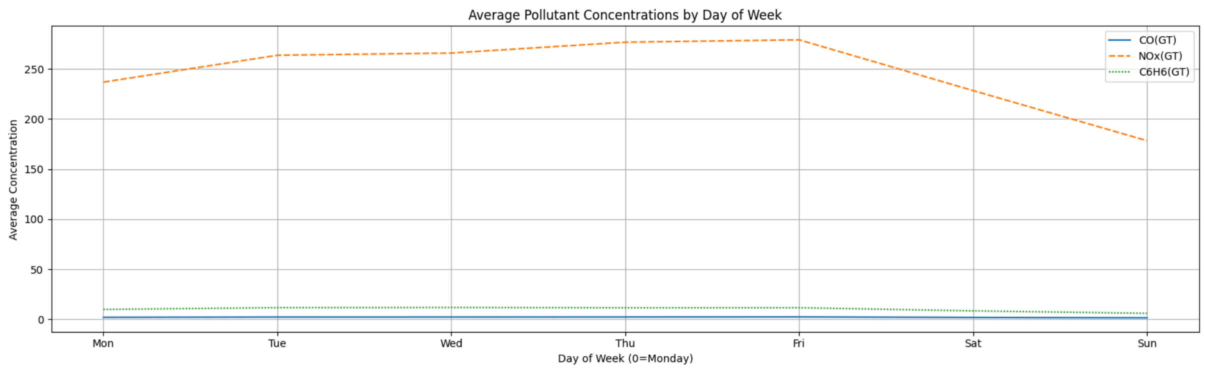
The highest pollutant levels occur during early morning and late evening, likely due to rush hours when traffic congestion is at its peak.
Notably, pollutant levels drop during late night and early mornings when traffic decreases and industrial activities slow down.



Weekly trends:

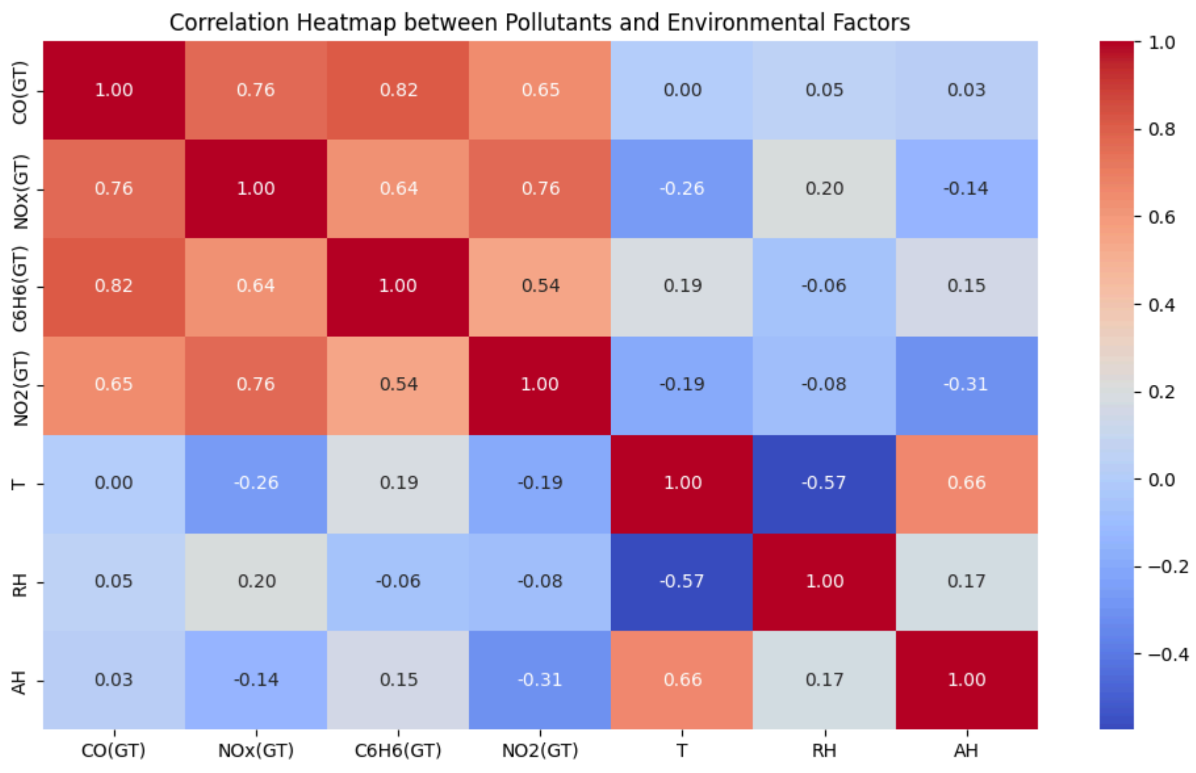
- Higher NOx trends on weekdays, than on weekends. This is consistent with the theory of vehicular emissions and their relation to traffic.
- CO and Benzene show minor variations but remain stable most of the time.

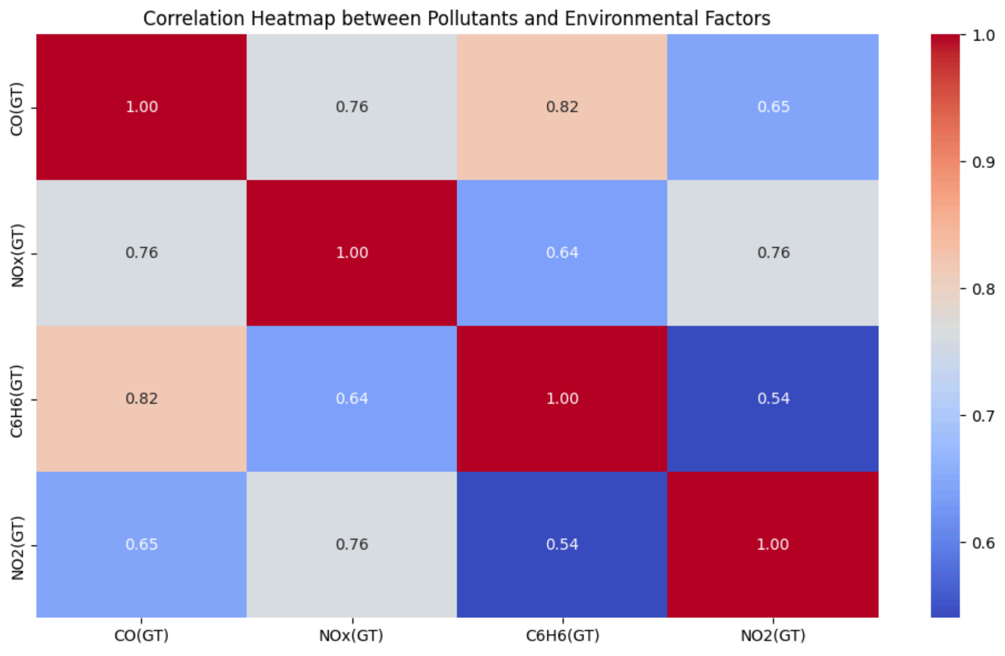
Weekends tend to have lower pollutant concentrations, reflecting a reduction in both traffic and industrial emissions.



3. Relationships between Pollutants

A correlation matrix shows:





The correlation heatmap shows positive correlation between CO, NOx, and Benzene. This means that they are not so independent of each other.
CO and NOx are associated with traffic emissions and thus are correlated.
The correlation between CO and Benzene is moderate, which may suggest that both are not so related and there might be other factors that influence the increase of benzene independently.
The correlation between NOx and Benzene is weaker, indicating that they might share some sources but are mostly influenced by different factors.

4. Potential Influencing Factors

Factor	Influence Level	Description
Traffic Volume	High	Strong rush hours signals in CO and NOx patterns
Day of the week	High	Lower pollutants on weekends indicate human related emissions, be it by traffic or Industrial activities.
Temperature/Humidity (Seasonality)	Medium	Can influence reactions or pollutant dispersion.

5. Implications for Modelling

- Time-Aware Modelling: Models like LSTM or ARIMA can be used to capture seasonality and trend components.
- Multivariate approach: Considering correlated pollutants together may enhance accuracy.
- Anomalies and Outliers: The pollutant levels might change because of some special events or some unusual weather patterns.
- Seasonality: Seasonality can play a huge role if we consider temperature a major feature.

Modeling approach and results

The goal of the modelling phase was to predict CO(GT), the concentration of CO at a given time. Given the time-series nature of the data, XGBoost was selected due to its high performance and flexibility in handling complex relationships and interactions between the features.

1. Modelling Strategy:
    - XGBoost was trained using the features created during the feature engineering phase.
    - The model was evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) as performance metrics.
    - Models such as Linear Regression and Random forest were also trained along with it.

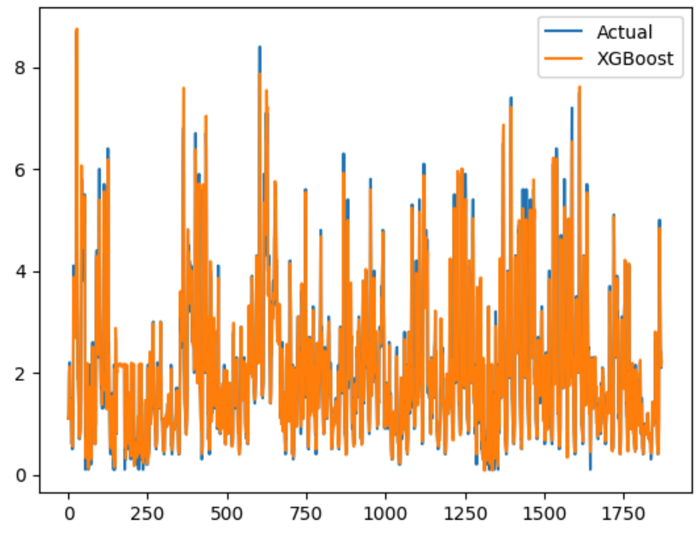
2. Model Training and Evaluation:
    - Training Process: The XGBoost model was trained using the training dataset, which comprised of 80% of the data, while the remaining 20% of the data was used for the testing phase.

    - Performance Metrics: The model was evaluated using MAE and RMSE.
      MAE: 0.084
      RMSE: 0.150

These results were then compared to the baseline model results, which predicted the previous value of CO for each time step. The XGBoost model outperformed the baseline model significantly, showing its effectiveness in predicting and handling complex datasets.

Linear Regression got great results but it was either because of data leakage or data overfitting.

3. Model Validation:
   The model was validated using the test split, the last 20% of the data.
   The plot for XGBoost vs Actual is as follows:



4. Integration with Kafka:
   The trained XGBoost model was integrated into the Kafka consumer script. This allowed the model to make real-time predictions on the incoming data streamed by the Kafka producer.

Conclusion and Limitations

Conclusion:
In this project, we successfully implemented a real-time air quality prediction system using Apache Kafka for data streaming and XGBoost for predicting CO concentrations. The Kafka producer and consumer pipeline efficiently handled the real-time flow of data, and the XGBoost model provided accurate predictions, outperforming the baseline model.

Key findings include:
- Temporal patterns, such as higher pollutant levels during specific hours and on weekdays.
- Correlation between different pollutants, suggesting shared environmental sources.

Limitations:
- While the project showed its feasibility of real-time air quality prediction, there are several limitations:
- Data Quality: The dataset had missing values and required preprocessing on an extensive level. If it is working on real-time data, it needs to have the data preprocessing and formatting done.
- Model Complexity: The model’s performance could potentially be improved with a more advanced model, such as LSTM (Long Short Term Memory), which is better suited for time-series data.
- Seasonal Data: The system works with historical data, but including the weather data could improve accuracy in dynamic environments.