Real-Time Air Quality Prediction with Apache Kafka

Executive Summary

This report presents an analysis of hourly air quality measurements collected in an Italian city over a one-year period from March 2004 to February 2005. The technical architecture for this project was designed to simulate a real-time streaming environment using the UCI Air Quality dataset through a single-node Kafka cluster implemented in KRaft mode, eliminating the need for ZooKeeper and relying on three key components: broker, producer, and consumer. The broker established the streaming connection and hosted a single topic for all sensor readings, with initial classpath configuration issues resolved by customizing the server.properties file. The producer replayed the UCI dataset row by row to simulate live ingestion, batching 100 rows per second and serializing them into JSON for readability and schema validation, while dynamically retrieving the dataset. The consumer subscribed to the topic, applied schema validation, converted missing values from -200 to NaN for standardization, and wrote validated rows in partitioned CSV files organized by month-year for efficient downstream processing.

Data Intelligence and Pattern Analysis revealed clear discrepancies between ground truth measurements and sensor data. While ground truth values remained relatively stable across most pollutants, with fluctuations mainly in NOx (GT) and NO2 (GT), sensor readings were far more volatile and often deviated significantly from the actual concentrations. Cyclical analyses highlighted that sensors not only exaggerated daily fluctuations but also reported higher values than ground truth, suggesting possible measurement errors. Correlation testing confirmed strong dependencies between certain pollutants, while decomposition analysis indicated that most pollutants followed stable seasonal and residual patterns, except NMHC (GT), which showed irregular variations. Anomaly detection further emphasized these differences, with NO2 (GT) anomalies emerging after February 2025, whereas NO2 sensor anomalies appeared earlier, underscoring sensor-specific inconsistencies.

Predictive analytics combined feature engineering with model evaluation to capture temporal patterns in the data. Seasonality was incorporated through one-hot and cyclical encoding, and three approaches were tested: baseline forward fill, Random Forest, and LSTM. Using RMSE, R^2, and MAE as metrics, Random Forest consistently outperformed the baseline except for NMHC, where low variance limited accuracy, while LSTM underperformed across pollutants. Surprisingly, the baseline worked well for CO and NO2, showing that simple methods can sometimes suffice. For long-term use, a scalable deployment and monitoring framework is recommended, leveraging Docker, cloud platforms, and Kafka for real-time streaming and reliable performance tracking.

Recommendations to scale this project even further include deploying Kafka on cloud-managed services to improve scalability, integrating multiple producers with additional datasets to gain more comprehensive

insights, and connecting the consumer pipeline directly to machine learning models such as Random Forest and LSTM for near-real-time predictions. Additionally, leveraging automated or augmented solutions to proactively regulate chemical concentrations can help translate data insights into actionable environmental sustainability measures.

Technical Architecture and Infrastructure Implementation

The technical architecture for this project was designed to simulate a real-time streaming environment using the UCI Air Quality dataset. To set up a data flow that manages streaming ingestion of sensor data, a single-node Kafka cluster was chosen. The infrastructure was implemented in KRaft mode (without external ZooKeeper), resulting in only three main components: broker, producer, and consumer.

Broker

The Kafka broker is crucial in establishing the streaming connection between the producer and consumer, even if the producer and consumer are unaware of each other. The broker was configured with one main topic, air_quality, used for all sensor readings. One main issue faced when setting up the broker was Classpath errors, even though the right binary source was extracted. This was resolved by creating a server.properties document in the correct directory with the needed requirements to ensure that Kafka can run smoothly and the Universally Unique Identifier (UUID) can be extracted, and the broker can function.

Producer

The producer was responsible for publishing messages to the air_quality topic. Instead of real-time sensors, the UCI dataset was replayed row by row to simulate live ingestion. The rows were sent out in 100 count each time with a 1-second delay so that the computational effort is lessened. Each record was serialized into JSON format to ensure readability and schema validation downstream. Furthermore, instead of downloading the dataset, to account for changes in the original dataset, fetch_ucirepo was used to retrieve data dynamically.

Consumer

The consumer subscribed to the air_quality topic, which was designed to prioritize both schema validation and reliable downstream storage. Data validation and cleaning were employed on the consumer side, so every row of data received underwent validation to verify that all 15 expected fields were present, and invalid messages were skipped with warnings logged. Furthermore, missing values in the original dataset were marked as -200, but to standardise formatting and understanding, these values were converted to NaN values.

The next challenge came in finding an intuitive way to store the data so that it can be utilised easily for further processing. To ensure efficient storage, messages were processed in batches of 100 before being written to partitioned CSV files. Partitioning was performed on a month-year basis derived from the Date

field. This ensures that even if the rows received are not sorted by date, the function would ensure that the rows are formatted and attributed to the right file. Logging was incorporated throughout to track ingestion health and highlight schema or format anomalies.

Data Intelligence and Pattern Analysis

As part of foundation visualization and advanced analytics, temporal analysis, cyclic trends analysis, correlation, and statistical significance analysis, autocorrelation and partial autocorrelation, decomposition analysis, and anomaly detection were performed on the data. It is to be noted that the NaN values in the dataset were filled with the mean of the column, which includes the mean of all the rows across all years. This ensures that the visualisations and analysis are minimally affected.

Temporal Analysis

The foundation visualization performed on this data mainly compared the effectiveness of the sensors in detecting the chemical concentrations as compared to the ground truth data.

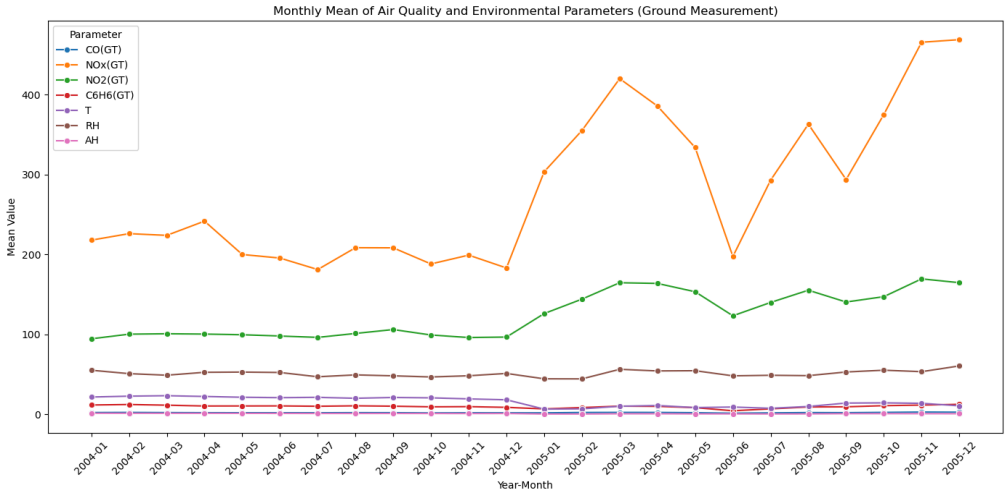


Figure 1: Temporal Analysis of Monthly Mean (Ground Measurement)

The monthly-year trends of this analysis for ground measurement generally show a constant level across all chemicals, except for NOx (GT) and NO2 (GT), which are demonstrated to fluctuate more, especially in 2025, compared to 2024.

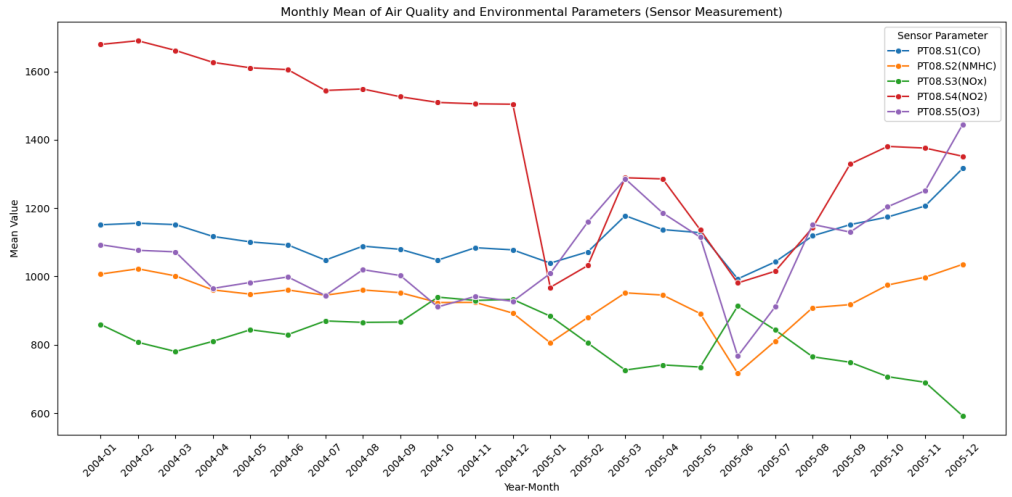


Figure 2: Temporal Analysis of Monthly Mean (Sensor Measurement)

In contrast, the corresponding sensor readings for the same year-month periods show substantial volatility across all measurements, suggesting that the sensors may not reliably reflect the true chemical concentrations. There is also a discrepancy in the year analysis between ground truth measurements and sensor readings, which further highlights the ineffectiveness of sensor readings across both types of analysis.

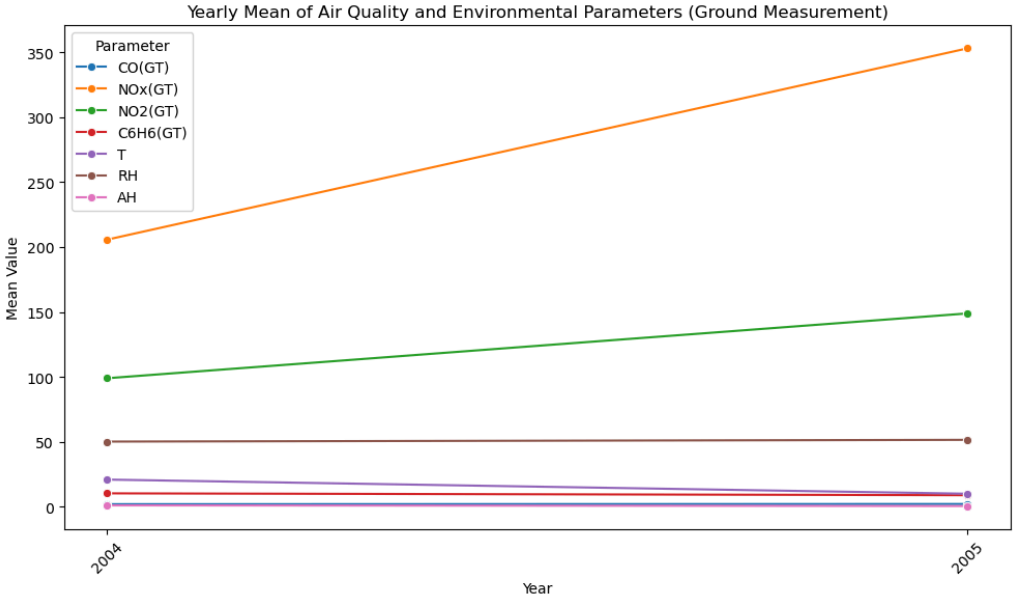


Figure 3: Temporal Analysis of Yearly Mean (Ground Measurement)

Comparing the yearly mean for ground measurement shows that there is a significant positive difference for NOx(GT) and NO2(GT).

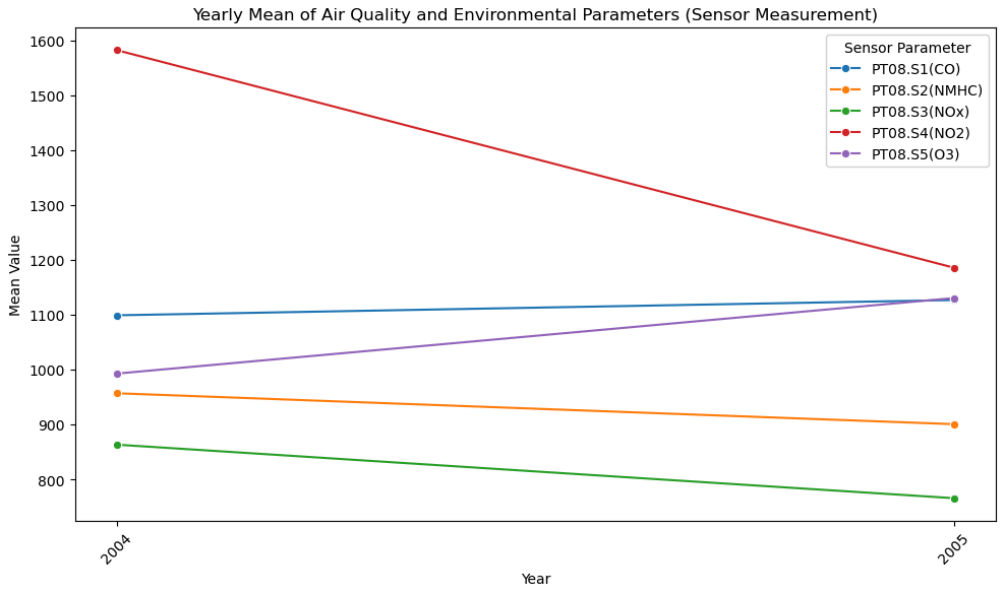


Figure 4: Temporal Analysis of Yearly Mean (Sensor Measurement)

A contrasting difference is observed for the sensors, with the reading for NO2 decreasing significantly. This volatility needs to be explored further.

Cyclic Trends

Daily cyclic trends were analyzed by taking the average per hour across the days.

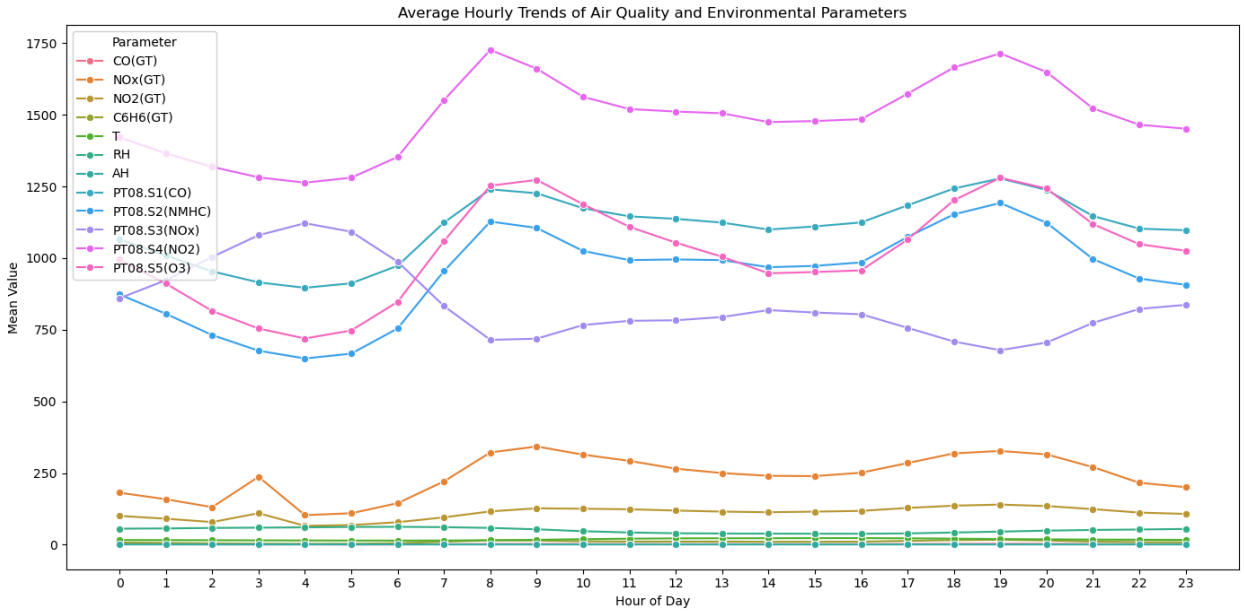


Figure 5: Cyclical Hourly Trends

NOx (GT) and NO2 (GT) ground truth measurements were more volatile, as expected, during the day as well. However, an interesting observation is that the sensor readings were not only significantly more

fluctuating but also significantly higher than the ground truth measurements, which could suggest that there might be a measurement error in these instruments.

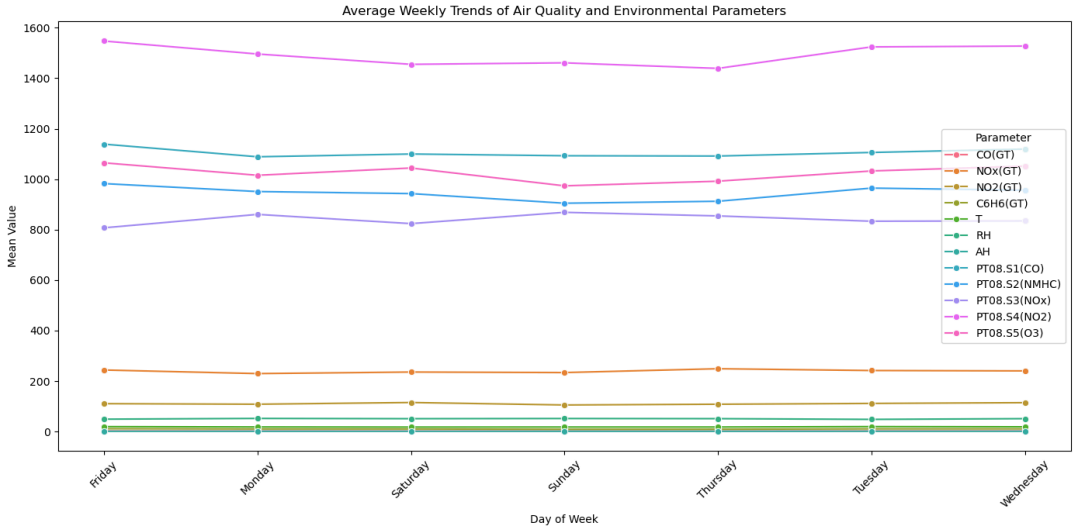


Figure 6: Cyclical Weekly Trends

To analyze the days as a whole, weekly cyclic analysis was also plotted by taking the average across the different weeks to see if the day makes a difference. It was surprising to note that the lines were most stable in this graph, showing that there is minimal difference across days, even for the fluctuating chemicals like NOx (GT) and NO2 (GT).

Correlation and Statistical Significance

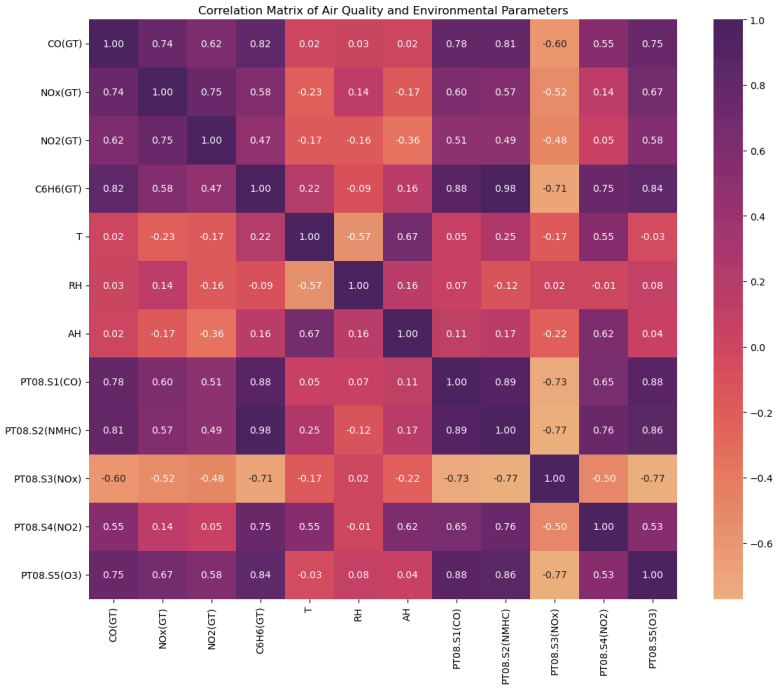


Figure 7: Correlation Analysis

To understand the data more from a statistical perspective, correlation and p-values were calculated. The sensors measuring CO and NHMC seemed to be highly correlated with many attributes on the table, including variables such as C6H6, which shows that there are dependencies. This analysis, therefore, gives insight into which pollutants or sensor readings influence each other. All the p-values are nearing 0 or are 0, which shows that the observed effect or relationship is highly statistically significant.

Autocorrelation and Partial Autocorrelation

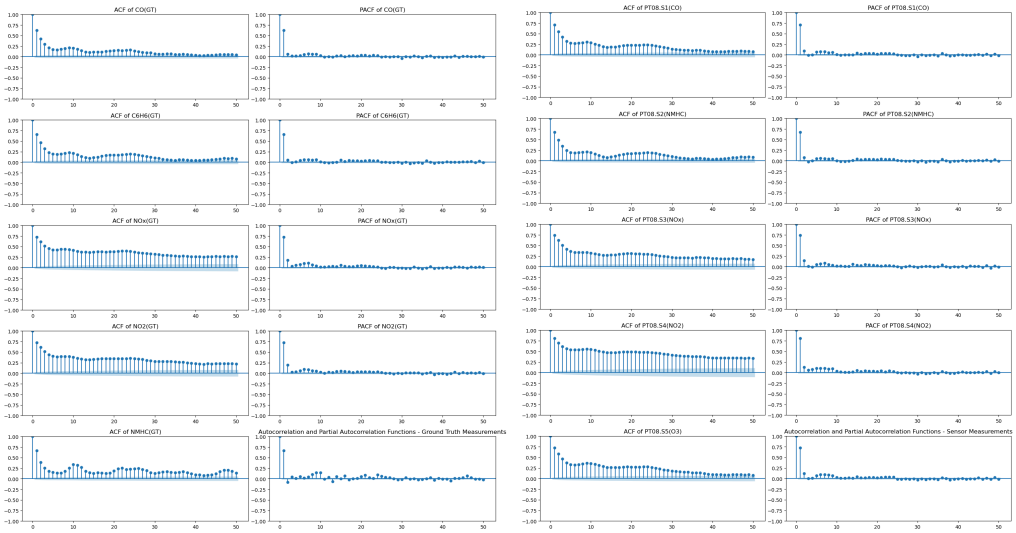
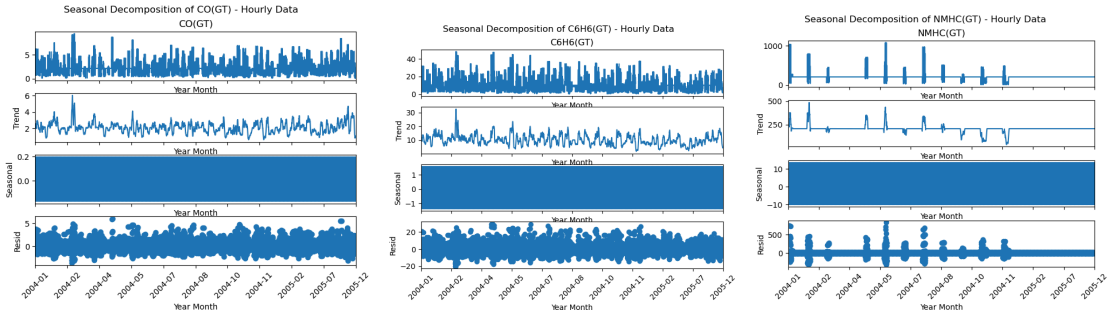


Figure 8: Autocorrelation and Partial Autocorrelation Analysis

The Autocorrelation Function shows the correlation between a time series and its lagged versions, while the Partial Autocorrelation Function shows the correlation between the time series and its lag, after removing the effects of shorter lags. For the ground truth measurements, there is a constant positive lag in their Autocorrelation Function that is more significant in the beginning. However, for their Partial Autocorrelation Function, the graph remains constant around the 0 axis with minimal deviations above and below, which highlights that there are no significant time lag differences. This holds for the sensor measurements as well.

Decomposition Analysis



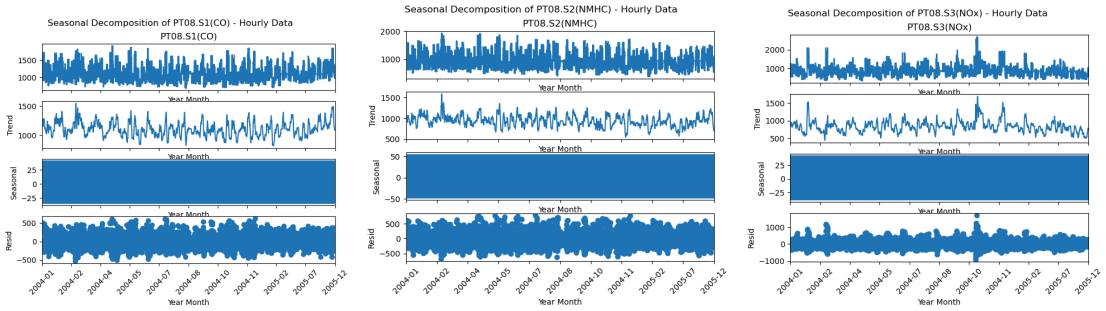


Figure 9: Decomposition Analysis

Decomposition analysis separates a time series into trend, seasonal, and residual components to better understand its underlying patterns. Seasonal and residual trends seemed to be constant with no change except for NMHC(GT), which had a residual pattern. This could be because NMHC(GT) experiences irregular fluctuations or short-term variations not explained by the overall trend or seasonality. In terms of the general trend, the chemicals had fluctuations across time, and this could be attributed to changes in environmental conditions, human activities, or other activities contributing to the pollution. For sensor measurements, Seasonal and Residual trends were not visible, and similar to the ground truth measurements, there were fluctuations in the general trend across time through peaks and troughs.

Anomaly Detection

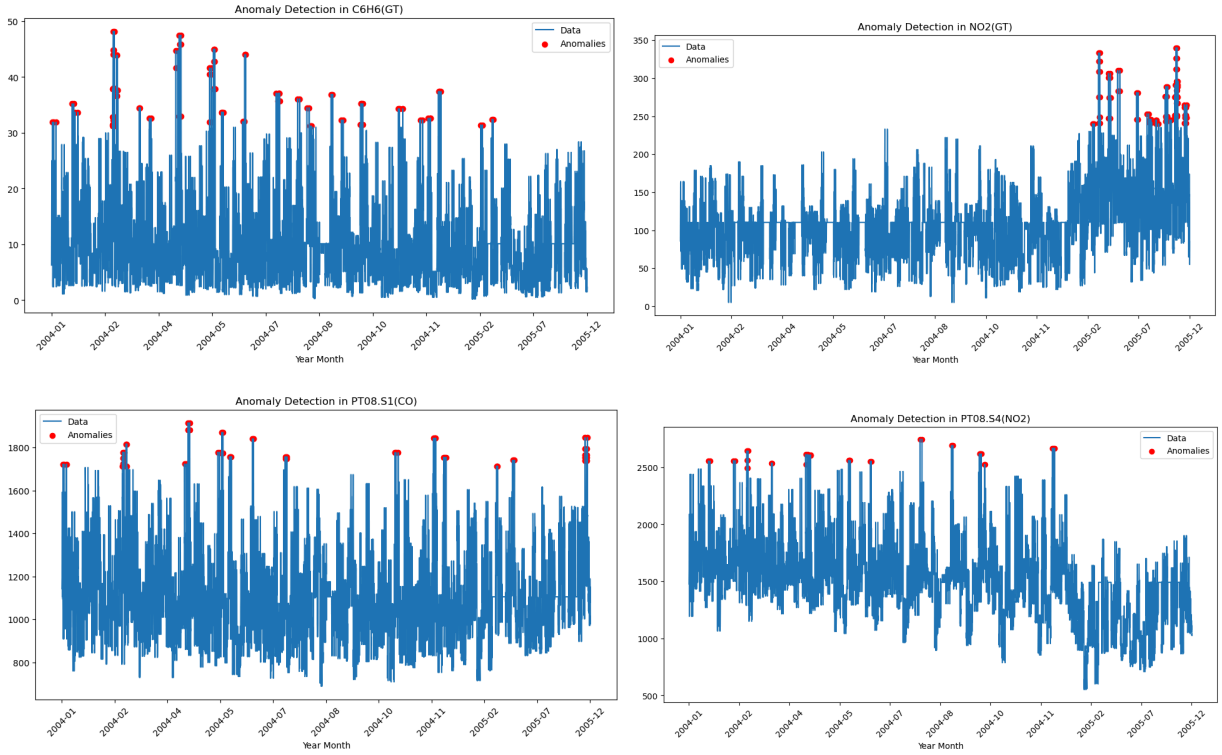


Figure 10: Anomaly Detection Analysis

For ground truth measurements, the anomalies detected were all on the higher extreme, and this could be because of sudden spikes in pollutant levels, which could have pushed the readings well above typical

values. Interestingly, while for all chemicals the anomalies were spread out across time, for NO2(GT), the anomalies arose only after February 2025. This suggests that for NO2(GT), a specific significant change or event starting around February 2025 affected the readings.

For sensor measurements, the trend of anomalies being significantly higher continued. However, interestingly, for the NO2 sensor, the anomalies showed up before the period of February 2025. This suggests that sensor-specific factors or local events may have caused anomalies ahead of the trend observed in the ground truth measurements.

Predictive Analytics and Model Performance

Feature Engineering

Date-related columns were generated based on the date columns to incorporate seasonality into the analysis. A new column called Seasons was also generated to aid with analysis, and this included using the Month column to group the rows into seasons and then applying one-hot encoding to change the columns into numeric ones for analysis. In addition, cyclical encoding was applied to both the Hour and Month variables using sine and cosine transformations. This ensures that the cyclical nature of time (e.g., 23:00 close to 00:00, December close to January) is preserved, enabling the models to better capture temporal patterns.

Predictive *Analytics*

For predictive analytics, we used three forms of measurement. Firstly, baseline measurement was performed using forward fill (ffill) as a simple imputation-based benchmark. Secondly, Random Forest was employed because it is an ensemble method that can handle non-linear relationships and is relatively robust to noise and overfitting, given the dynamic changes in the measurements as seen above. Lastly, the LSTM algorithm was used to model the sequential and temporal dependencies in the time-series data, leveraging its ability to retain information over long sequences and capture seasonality or lag effects that traditional models may miss. To evaluate model performance, three metrics were selected: Root Mean Squared Error (RMSE) to penalize large errors more heavily, R^2 (coefficient of determination) to assess the explained variance, and Mean Absolute Error (MAE) to measure average prediction accuracy in an interpretable way.

	Baseline_RMSE	Baseline_R2	Baseline_MAE	RandomForest_RMSE	RandomForest_R2	RandomForest_MAE	LSTM_RMSE	LSTM_R2	LSTM_MAE
CO(GT)	1.141277	0.243215	0.684330	0.804571	6.632457e-01	0.538787	1.287906	1.379575e-01	0.931257
C6H6(GT)	33.390092	0.313139	3.736177	0.029435	9.999762e-01	0.012937	6.073926	-1.376585e-02	4.559173
NMHC(GT)	3126.714984	0.324911	12.280400	63.150930	-4.936954e+30	41.548760	128.514971	-2.044594e+31	111.468115
NOx(GT)	17431.584722	0.455097	78.551155	142.815136	5.778612e-01	96.461562	286.766760	-7.006490e-01	208.082472
NO2(GT)	1030.288784	0.446477	21.128452	40.870209	3.507364e-01	29.553529	77.727528	-1.347183e+00	63.834798

Figure 10: Summary of Model Performance

Overall, the results indicate that Random Forest consistently outperforms the baseline across most pollutants, except for NMHC, where the outcome first suggests a possible calculation or implementation issue. However, after exploring the range of NHMC values, the 25th, 50th, and 75th percentiles were all

the same value, which suggests there is less variation for the model to be effective in predicting. In contrast, the LSTM model underperforms for all pollutants, which may be attributed to factors such as poor hyperparameter tuning or the need for further feature engineering methods, which can be explored. Lastly, surprisingly, the baseline model performs reasonably well for certain pollutants, such as CO and NO2, highlighting that simple averaging methods are sometimes sufficient to capture underlying patterns in the data, especially since these methods are also much quicker and cost-effective.

Based on the initial analysis of the data and the predictive analytics, a production deployment and monitoring framework has to be established to maintain the relevance of the insights. The deployment of predictive models into production should follow a pipeline-based approach that ensures reproducibility, scalability, and maintainability. Tools like Docker allow for containerization, and platforms like AWS allow for the system to be maintained in the cloud.

Strategic Conclusions and Future Enhancements

To conclude, this project successfully demonstrated how streaming frameworks like Kafka can be leveraged to simulate and manage environmental sensor data flows. However, limitations of this analysis would include data source constraints, context-based information, and infrastructure complexity. The UCI Air Quality dataset, while useful for simulation, lacked real-time variability and contained missing or placeholder values, which did not provide reasoning as to why the values were missing, and this could affect the analysis. Furthermore, the variability in the chemical measurements was pretty volatile and was different from the sensors, but the lack of context information regarding the happenings in the city or the tools themselves hinders the completeness of the anomalies detected. Lastly, technically, setting up the Kafka environment locally and configuring and maintaining brokers, producers, and consumers was challenging at first. This shows that while the pipeline was effective for prototyping, scaling such a system into production would require more robust infrastructure, automated error handling, and integration with real-time, multi-source data.

This leads to some recommendations to scale this project. One would be to deploy Kafka on cloud-managed services to improve scalability and use multiple producers with other datasets to gain a more comprehensive insight regarding the readings and the prediction. In terms of model integration, it would be more effective to connect the consumer pipeline directly to machine learning workflows (e.g., Random Forest, LSTM) for near real-time predictions and proactive interventions. Lastly, the data and insights need to be utilised to drive proactive measures and policy initiatives. So, after figuring out the model that the organisation wants to use for prediction, enabling automated or augmented solutions to proactively regulate the chemical concentrations for environmental sustainability would be helpful.