

Project 2:

Air Quality analysis using Spark

Group Members:

Satya Datta Jupalli
Krishna Teja Rayapudi
Jyotika Koneru
Narayana Datta Jupalli
Sudhish Cherukuri

Link:https://github.com/Scheruk1701/air_quality_analysis_spark.git

Index

1. Introduction
2. Dataset Overview
3. Pipeline Architecture
 - Section 1: Data Ingestion
 - Section 2: Data Transformation & Enrichment
 - Section 3: SQL-Based Insights
 - Section 4: ML Forecasting Module
 - Section 5: Unified Pipeline & Dashboard Visualization
4. Visualization Samples
5. Technologies Used
6. Enhancements & Recommendations
7. Tasks & Difficulties Faced
8. outcomes

1. Introduction

Air pollution, especially in urban areas, has emerged as one of the critical global concerns affecting public health and quality of life. Prolonged exposure to pollutants like PM2.5 can lead to respiratory issues, cardiovascular diseases, and a decline in the general well-being of populations. Thus, timely monitoring and forecasting of air quality is essential for effective interventions.

The goal of this project is to develop a comprehensive, end-to-end air quality monitoring and forecasting system. This solution utilizes Apache Spark to handle large-scale data processing efficiently and applies machine learning techniques to predict pollutant levels. The project is structured around a modular pipeline that includes:

- Data ingestion from raw sensor outputs.
- Data cleaning, transformation, and enrichment to prepare it for analysis.
- Analytical queries using Spark SQL to extract meaningful patterns.
- Machine learning models to predict PM2.5 concentration levels.
- Dashboard visualizations to present real-time and historical insights to stakeholders

2. Dataset Overview

The dataset used for this project comprises time-series sensor data capturing environmental air quality metrics from various regions. Each data point records key attributes such as particulate matter (PM2.5) concentration, temperature, humidity, and a timestamp, enabling both temporal and spatial analysis.

The raw data originates from simulated or publicly available air quality sources, and it has been processed through a multistage pipeline to enhance its utility for both analytical and machine learning purposes.

1. Raw Data Features:

- timestamp: The exact time the observation was recorded.

- region: Geographical identifier for the sensor location.
- pm25: Concentration of fine particulate matter (PM2.5) in micrograms per cubic meter.
- temperature: Ambient air temperature in Celsius.
- humidity: Relative humidity percentage.

2. Engineered Features:

- pm25_zscore, temperature_zscore, humidity_zscore: Standardized values to enable comparability and normalization.
- AQI_Category: Categorized pollutant levels based on PM2.5 thresholds (Good, Moderate, Unhealthy).
- Rolling Averages / Lag Features (optional): These were explored in ML experimentation for time-series modeling.

3. Data Storage Formats:

- CSV: Used throughout the pipeline for intermediate outputs and task-specific deliverables.
- Parquet: Employed for optimized query performance and ML model ingestion.

These features form the backbone of the predictive and analytical capabilities of the pipeline. As the data progresses through ingestion, cleaning, transformation, and modeling, these columns are filtered, enriched, and reformatted as needed for each stage.

3. Technologies Used

- Apache Spark (SQL, MLlib)
- Python (Pandas, Plotly, Matplotlib)
- Jupyter Notebooks for integration
- Optional Kafka + Docker (for real-time streaming support)

4. Pipeline Architecture

The system is structured into modular stages that together create a seamless data flow from raw ingestion to actionable insights:

Section 1: Data Ingestion

- Simulates near-real-time ingestion via TCP (batch for prototype).
- Parses timestamps, removes noise, and merges separate metrics into single unified rows.
- Joins external temperature/humidity data if necessary.
- Output: Cleaned and joined DataFrame ready for transformation.

TCP file client port connect to 9999

```
@satyadatta2001 →/workspaces/air_quality_analysis_spark (master) $ python ingestion/tcp_log_file_streaming_server.py
TCP server listening on localhost:9999...
Waiting for new client...
```

```

@satyadatta2001 → /workspaces/air_quality_analysis_spark (master) $ python section1_ingestion.py
25/04/29 22:09:16 WARN Utils: Your hostname, codespaces-cbd5c9 resolves to a loopback address: 127.0.0.1; using 10.0.4.193 instead (on interface eth0)
25/04/29 22:09:16 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
25/04/29 22:09:16 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
25/04/29 22:09:18 WARN TextSocketSourceProvider: The socket source should not be used for production applications! It does not support recovery.
25/04/29 22:09:19 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not supported in streaming DataFrames/Datasets and will be disabled.
Streaming started. Batches are being saved. Press Ctrl+C to stop.

```

```

1 timestamp,region,pm25,temperature,humidity,location,,20.27,33.19
2 2024-12-31T22:30:00.000Z,New Delhi-8118,204.0,22.87,58.82
3 2025-01-01T02:30:00.000Z,New Delhi-8118,165.0,29.45,40.88
4 2025-01-01T06:30:00.000Z,New Delhi-8118,188.0,29.9,36.24
5 2025-01-01T10:30:00.000Z,New Delhi-8118,202.0,34.43,56.93
6 2025-01-01T14:30:00.000Z,New Delhi-8118,196.0,27.04,53.04
7 2025-01-01T18:30:00.000Z,New Delhi-8118,200.0,30.31,36.33
8 2025-01-01T21:30:00.000Z,New Delhi-8118,173.0,32.82,61.44
9 2025-01-02T01:30:00.000Z,New Delhi-8118,153.0,32.49,35.39
10 2025-01-02T05:30:00.000Z,New Delhi-8118,204.0,21.89,43.39
11 2025-01-02T09:30:00.000Z,New Delhi-8118,213.0,32.48,47.74
12 2025-01-02T13:30:00.000Z,New Delhi-8118,217.0,22.79,42.01
13 2025-01-02T17:30:00.000Z,New Delhi-8118,210.0,27.16,36.61
14 2025-01-02T20:30:00.000Z,US Diplomatic Post: New Delhi-8118,213.0,26.25,38.05
15 2025-01-03T00:30:00.000Z,US Diplomatic Post: New Delhi-8118,-999.0,20.93,54.48
16 2025-01-03T04:30:00.000Z,US Diplomatic Post: New Delhi-8118,210.0,22.05,58.72
17 2025-01-03T08:30:00.000Z,US Diplomatic Post: New Delhi-8118,382.0,24.54,60.51
18 2025-01-03T12:30:00.000Z,US Diplomatic Post: New Delhi-8118,213.0,33.63,31.25
19 2025-01-03T16:30:00.000Z,US Diplomatic Post: New Delhi-8118,345.0,21.82,31.61

```

Section 2: Data Transformation & Enrichment

- Handles missing values and outliers via capping/imputation.
- Normalizes features using z-score scaling.
- Computes hourly/daily aggregations.
- Creates lag, rolling average, and rate-of-change features.
- Output: Transformed dataset with analytical/ML-ready structure.

```

  output_task2
    cleaned_data
      _SUCCESS
      task2_cleaned_data.csv
      task2_cleaned_data.parquet
    daily_aggregations
      _SUCCESS
      task2_daily_aggregations.csv
      task2_daily_aggregations.parquet
    enhanced_data
      _SUCCESS
      task2_enhanced_data.csv
      task2_enhanced_data.parquet
    hourly_trends
      _SUCCESS
      task2_hourly_trends.csv
      task2_hourly_trends.parquet

```

Cleaned Data.csv

```

output_task2 > cleaned_data > task2_cleaned_data.csv
1  timestamp,region,pm25,temperature,humidity
2  2024-12-31 22:30:00+00:00,New Delhi-8118,204.0,22.87,58.82
3  2025-01-01 02:30:00+00:00,New Delhi-8118,165.0,29.45,40.88
4  2025-01-01 06:30:00+00:00,New Delhi-8118,188.0,29.9,36.24
5  2025-01-01 10:30:00+00:00,New Delhi-8118,202.0,34.43,56.93
6  2025-01-01 14:30:00+00:00,New Delhi-8118,196.0,27.04,53.04
7  2025-01-01 18:30:00+00:00,New Delhi-8118,200.0,30.31,36.33
8  2025-01-01 21:30:00+00:00,New Delhi-8118,173.0,32.82,61.44
9  2025-01-02 01:30:00+00:00,New Delhi-8118,153.0,32.49,35.39
10 2025-01-02 05:30:00+00:00,New Delhi-8118,204.0,21.89,43.39
11 2025-01-02 09:30:00+00:00,New Delhi-8118,213.0,32.48,47.74
12 2025-01-02 13:30:00+00:00,New Delhi-8118,217.0,22.79,42.01
13 2025-01-02 17:30:00+00:00,New Delhi-8118,210.0,27.16,36.61
14 2025-01-02 20:30:00+00:00,US Diplomatic Post: New Delhi-8118,213.0,26.25,38.05
15 2025-01-03 00:30:00+00:00,US Diplomatic Post: New Delhi-8118,-999.0,20.93,54.48
16 2025-01-03 04:30:00+00:00,US Diplomatic Post: New Delhi-8118,210.0,22.05,58.72
17 2025-01-03 08:30:00+00:00,US Diplomatic Post: New Delhi-8118,382.0,24.54,60.51
18 2025-01-03 12:30:00+00:00,US Diplomatic Post: New Delhi-8118,213.0,33.63,31.25
19 2025-01-03 16:30:00+00:00,US Diplomatic Post: New Delhi-8118,345.0,21.82,31.61

```


Daily.csv which will have all the daily limit

```
output_task2 > daily_aggregations > task2_daily_aggregations.csv
1 date,region,pm25,temperature,humidity
2 2024-12-31,New Delhi-8118,278.2,25.27,49.42
3 2025-01-01,New Delhi-8118,187.54,29.12,46.9
4 2025-01-02,New Delhi-8118,207.11,27.83,47.02
5 2025-01-02,US Diplomatic Post: New Delhi-8118,210.8,28.82,38.11
6 2025-01-03,US Diplomatic Post: New Delhi-8118,158.46,27.54,49.19
7 2025-01-04,New Delhi-8118,-504.8,24.36,57.47
8 2025-01-04,US Diplomatic Post: New Delhi-8118,296.89,26.71,51.91
9 2025-01-05,New Delhi-8118,-999.0,27.34,46.91
10 2025-01-05,US Diplomatic Post: New Delhi-8118,-999.0,29.34,49.44
11 2025-01-06,US Diplomatic Post: New Delhi-8118,-999.0,29.42,45.64
12 2025-01-07,US Diplomatic Post: New Delhi-8118,-999.0,28.09,47.85
13 2025-01-08,US Diplomatic Post: New Delhi-8118,-131.21,27.74,50.24
14 2025-01-09,US Diplomatic Post: New Delhi-8118,292.67,28.05,49.12
15 2025-01-10,US Diplomatic Post: New Delhi-8118,165.25,25.0,53.22
16 2025-01-11,US Diplomatic Post: New Delhi-8118,157.42,28.66,50.91
17 2025-01-12,New Delhi-8118,139.4,26.4,46.84
18 2025-01-12,US Diplomatic Post: New Delhi-8118,157.95,26.48,55.98
19 2025-01-13,New Delhi-8118,120.95,27.31,47.03
```

Hourly.csv

```
output_task2 > hourly_trends > task2_hourly_trends.csv
1 timestamp,region,pm25,temperature,humidity,pm25_zscore,temperature_zscore,humidity_zscore,date,hour,pm25_rolling
2 2024-12-31 19:30:00+00:00,New Delhi-8118,371.0,32.38,59.61,1.43,1.11,0.87,2024-12-31,19,371.0,,
3 2024-12-31 20:30:00+00:00,New Delhi-8118,343.0,27.45,38.87,1.3,-0.02,-0.95,2024-12-31,20,357.0,371.0,-28.0
4 2024-12-31 21:30:00+00:00,New Delhi-8118,286.0,23.27,33.96,1.04,-0.97,-1.38,2024-12-31,21,333.33,343.0,-57.0
5 2024-12-31 22:30:00+00:00,New Delhi-8118,204.0,22.87,58.82,0.66,-1.06,0.8,2024-12-31,22,277.67,286.0,-82.0
6 2024-12-31 23:30:00+00:00,New Delhi-8118,187.0,20.36,55.84,0.58,-1.63,0.54,2024-12-31,23,225.67,204.0,-17.0
7 2025-01-01 00:30:00+00:00,New Delhi-8118,189.0,22.57,65.3,0.59,-1.13,1.37,2025-01-01,0,193.33,187.0,2.0
8 2025-01-01 01:30:00+00:00,New Delhi-8118,185.0,27.03,42.15,0.58,-0.11,-0.66,2025-01-01,1,187.0,189.0,-4.0
9 2025-01-01 02:30:00+00:00,New Delhi-8118,165.0,29.45,40.88,0.48,0.44,-0.77,2025-01-01,2,179.67,185.0,-20.0
10 2025-01-01 03:30:00+00:00,New Delhi-8118,154.0,30.02,44.47,0.43,0.57,-0.46,2025-01-01,3,168.0,165.0,-11.0
11 2025-01-01 04:30:00+00:00,New Delhi-8118,170.0,22.15,30.81,0.51,-1.23,-1.65,2025-01-01,4,163.0,154.0,16.0
12 2025-01-01 05:30:00+00:00,New Delhi-8118,193.0,21.52,44.75,0.61,-1.37,-0.43,2025-01-01,5,172.33,170.0,23.0
13 2025-01-01 06:30:00+00:00,New Delhi-8118,188.0,29.9,36.24,0.59,0.54,-1.18,2025-01-01,6,183.67,193.0,-5.0
14 2025-01-01 07:30:00+00:00,New Delhi-8118,176.0,34.9,50.81,0.53,1.68,0.1,2025-01-01,7,185.67,188.0,-12.0
15 2025-01-01 08:30:00+00:00,New Delhi-8118,202.0,29.67,34.61,0.65,0.49,-1.32,2025-01-01,8,188.67,176.0,26.0
16 2025-01-01 09:30:00+00:00,New Delhi-8118,196.0,27.56,67.78,0.63,0.01,1.59,2025-01-01,9,191.33,202.0,-6.0
17 2025-01-01 10:30:00+00:00,New Delhi-8118,202.0,34.43,56.93,0.65,1.58,0.64,2025-01-01,10,200.0,196.0,6.0
18 2025-01-01 11:30:00+00:00,New Delhi-8118,204.0,25.61,34.07,0.66,-0.44,-1.37,2025-01-01,11,200.67,202.0,2.0
```


Enhanced csv

```
output_task2 > enhanced_data >  task2_enhanced_data.csv
 1  timestamp,region,pm25,temperature,humidity,pm25_zscore,temperature_zscore,humidity_zscore
 2  2024-12-31 22:30:00+00:00,New Delhi-8118,204.0,22.87,58.82,0.66,-1.06,0.8
 3  2025-01-01 02:30:00+00:00,New Delhi-8118,165.0,29.45,40.88,0.48,0.44,-0.77
 4  2025-01-01 06:30:00+00:00,New Delhi-8118,188.0,29.9,36.24,0.59,0.54,-1.18
 5  2025-01-01 10:30:00+00:00,New Delhi-8118,202.0,34.43,56.93,0.65,1.58,0.64
 6  2025-01-01 14:30:00+00:00,New Delhi-8118,196.0,27.04,53.04,0.63,-0.11,0.29
 7  2025-01-01 18:30:00+00:00,New Delhi-8118,200.0,30.31,36.33,0.64,0.64,-1.17
 8  2025-01-01 21:30:00+00:00,New Delhi-8118,173.0,32.82,61.44,0.52,1.21,1.03
 9  2025-01-02 01:30:00+00:00,New Delhi-8118,153.0,32.49,35.39,0.43,1.13,-1.25
10  2025-01-02 05:30:00+00:00,New Delhi-8118,204.0,21.89,43.39,0.66,-1.29,-0.55
11  2025-01-02 09:30:00+00:00,New Delhi-8118,213.0,32.48,47.74,0.7,1.13,-0.17
12  2025-01-02 13:30:00+00:00,New Delhi-8118,217.0,22.79,42.01,0.72,-1.08,-0.67
13  2025-01-02 17:30:00+00:00,New Delhi-8118,210.0,27.16,36.61,0.69,-0.08,-1.14
14  2025-01-02 20:30:00+00:00,US Diplomatic Post: New Delhi-8118,213.0,26.25,38.05,0.7,-0.29,-1.02
15  2025-01-03 00:30:00+00:00,US Diplomatic Post: New Delhi-8118,-999.0,20.93,54.48,-4.86,-1.5,0.42
16  2025-01-03 04:30:00+00:00,US Diplomatic Post: New Delhi-8118,210.0,22.05,58.72,0.69,-1.25,0.79
17  2025-01-03 08:30:00+00:00,US Diplomatic Post: New Delhi-8118,382.0,24.54,60.51,1.48,-0.68,0.95
18  2025-01-03 12:30:00+00:00,US Diplomatic Post: New Delhi-8118,213.0,33.63,31.25,0.7,1.39,-1.61
19  2025-01-03 16:30:00+00:00,US Diplomatic Post: New Delhi-8118,345.0,21.82,31.61,1.31,-1.3,-1.58
```

Section 3: SQL-Based Insights

- Registers data as Spark SQL views.
- Executes queries to identify trends, pollution spikes, and AQI classifications.
- Uses window functions (e.g., LAG, ROW_NUMBER) for trend detection.
- Defines UDFs to tag AQI categories.

▼ output_task3	●
▼ aqi_classification	●
≡ _SUCCESS	U
≡ ._SUCCESS.crc	U
≡ .part-00000-82c2e5ea-2b0b-4c5a-9428-1c127d215b7d-c...	U
≡ part-00000-82c2e5ea-2b0b-4c5a-9428-1c127d215b7d-c...	U
▼ highest_avg_pm25	●
≡ _SUCCESS	U
≡ ._SUCCESS.crc	U
≡ .part-00000-41033dc7-cf56-4f1b-9730-4a20b201659a-c0...	U
≡ part-00000-41033dc7-cf56-4f1b-9730-4a20b201659a-c0...	U
▼ peak_pollution_intervals	●
≡ _SUCCESS	U
≡ ._SUCCESS.crc	U
≡ .part-00000-65bd2355-d7ee-4e94-8a8b-b8caf162e362-c...	U
≡ part-00000-65bd2355-d7ee-4e94-8a8b-b8caf162e362-c0...	U
▼ pm25_trend_increase	●
≡ _SUCCESS	U
≡ ._SUCCESS.crc	U
≡ .part-00000-f6495347-fe68-4b52-9af0-70b389a95110-c0...	U
≡ part-00000-f6495347-fe68-4b52-9af0-70b389a95110-c0...	U

Section 4: ML Forecasting Module

- Focused on PM2.5 regression using Random Forest.
- Uses only **temperature** and **humidity** for model input.
- Train-test split with cross-validation and tuning.
- Outputs RMSE, R^2 scores and saves predictions to CSV.

Section 5: Unified Pipeline & Dashboard Visualization

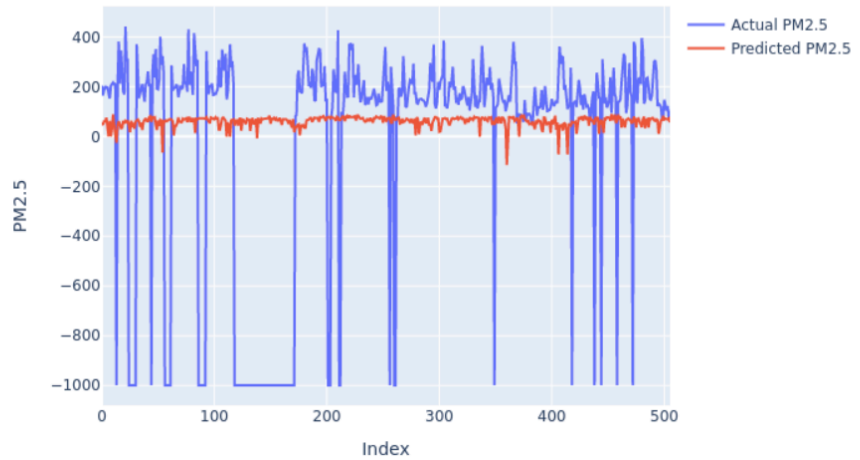
- Integrates all Spark stages into a reproducible pipeline script.
- Dashboards created with Plotly/Matplotlib:
 - Line charts for actual vs predicted PM2.5.
 - Event timelines for AQI breaches.
 - AQI breakdown (pie/bar charts).
 - Correlation heatmaps.
- Stores results in CSV/Parquet/PostgreSQL as needed.
- Can be expanded to support real-time Kafka ingestion and a Grafana dashboard.

1	timestamp	region	pm25	temperature	humidity	predicted_pm25
2	2024-12-31 22:30:00+00:00	New Delhi-8118	204.0	22.87	58.82	55.19101889906952
3	2025-01-01 02:30:00+00:00	New Delhi-8118	165.0	29.45	40.88	46.56068468275137
4	2025-01-01 06:30:00+00:00	New Delhi-8118	188.0	29.9	36.24	63.602203154743776
5	2025-01-01 10:30:00+00:00	New Delhi-8118	202.0	34.43	56.93	57.98502465525824
6	2025-01-01 14:30:00+00:00	New Delhi-8118	196.0	27.04	53.04	75.92038225679174
7	2025-01-01 18:30:00+00:00	New Delhi-8118	200.0	30.31	36.33	45.70201141527351
8	2025-01-01 21:30:00+00:00	New Delhi-8118	173.0	32.82	61.44	0.2733689375301547
9	2025-01-02 01:30:00+00:00	New Delhi-8118	153.0	32.49	35.39	63.05712406187656
10	2025-01-02 05:30:00+00:00	New Delhi-8118	204.0	21.89	43.39	0.2733689375301547
11	2025-01-02 09:30:00+00:00	New Delhi-8118	213.0	32.48	47.74	37.699088457044
12	2025-01-02 13:30:00+00:00	New Delhi-8118	217.0	22.79	42.01	89.07091946641829
13	2025-01-02 17:30:00+00:00	New Delhi-8118	210.0	27.16	36.61	40.996307217391504

5. Visualization Samples

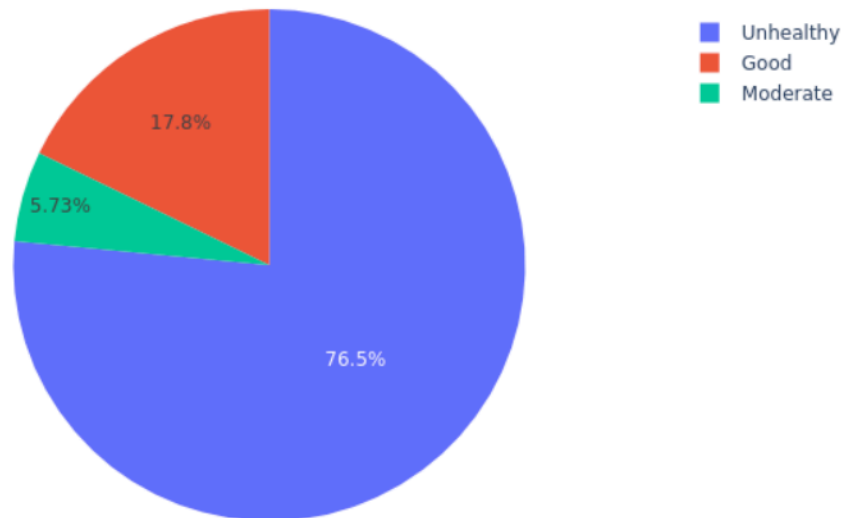
- **Time Series:** Actual vs Predicted PM2.5 trends.

Actual vs Predicted PM2.5 Over Time



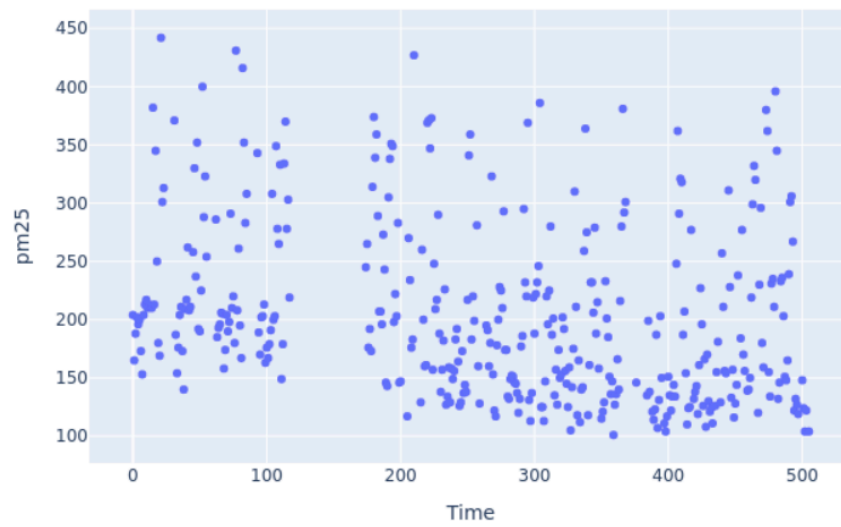
-
- **AQI Category Breakdown:** Pie chart showing percentages of Good, Moderate, Unhealthy.

AQI Classification Breakdown

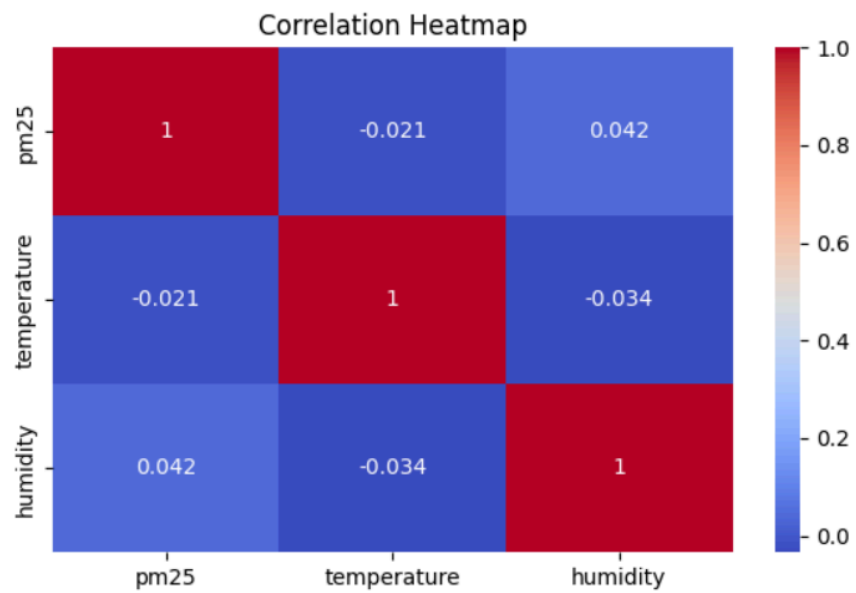


-
- **Spike Timeline:** Dates exceeding safe PM2.5 limits highlighted.

PM2.5 Spikes > 100



- **Correlation Matrix:** Pearson heatmap among PM2.5, temp, and humidity.



6. Enhancements & Recommendations

- **Integrate Kafka for Real-Time Data Feed:**
Instead of batch-mode ingestion, use Apache Kafka to stream live sensor readings into Spark Structured Streaming for low-latency, continuous updates.
- **Add More Predictive Features:**
Expand the dataset by including additional environmental metrics like:
 - Wind speed
 - Nitrogen Dioxide (NO₂) levels
 - Carbon Monoxide (CO) levels
 - Pressure, rainfall, or solar radiationThese variables can improve model robustness and forecasting accuracy.
- **Deploy ML Model with Structured Streaming:**
Adapt the trained model to perform live scoring (predictions) as data streams in from Kafka, enabling immediate alerts for air quality deterioration.
- **Automate Dashboard Updates:**
Build a lightweight REST API server (e.g., Flask or FastAPI) that continuously feeds predictions and processed data into your dashboard without manual refreshes.
- **Advanced Feature Engineering:**
Introduce:
 - Seasonal decomposition (seasonality patterns in PM2.5)
 - Holiday or festival effects (known to influence pollution)
 - Interaction features (e.g., PM2.5 × humidity)
- **Model Ensemble Approach:**
Instead of relying on a single Random Forest model, combine multiple models (e.g., Random Forest + Gradient Boosted Trees + XGBoost) for more stable and accurate results.

- Alert System for Critical AQI Levels:
Create a real-time email/SMS alerting system whenever PM2.5 crosses hazardous thresholds, notifying authorities or residents instantly.
- Deploy Full System via Docker Compose:
Bundle Spark, Kafka, PostgreSQL, and a Dashboard server using Docker Compose for seamless local or cloud deployment.
- Dashboard Enhancement Ideas:
 - Add drill-down filters by region or date.
 - Introduce interactive prediction sliders (simulate future scenarios).
 - Compare historical pollution data vs real-time.
- Long-Term Storage:
Archive old sensor data in cloud services like AWS S3 or Azure Blob Storage for future retraining and analysis.
- Scalability Considerations:
Optimize Spark configurations (memory management, caching) to handle millions of records per day if system scales up.

7. Tasks & Difficulties Faced

Tasks Executed:

- Simulated ingestion from TCP stream
- Data cleaning and schema normalization
- Feature engineering with lag/rolling stats
- SQL-based AQI classification
- Regression model training and hyperparameter tuning
- Dashboard design and export to reports

Challenges Encountered:

- Handling timestamp conversions and watermarks
- Managing missing data across time windows

- VectorAssembler issues with nulls during ML stage
- Maintaining schema consistency between stages
- Setting up Kafka and Docker locally (optional)
- Visualization rendering limitations in notebooks

8. Outcome

This system demonstrates a functional, scalable solution for air quality monitoring and forecasting. The integration of batch analytics and ML forecasts into a single pipeline enhances transparency and data-driven decision-making. Key capabilities include:

- Real-time and batch ingestion of environmental data.
- Sophisticated feature engineering and SQL-based analytical insight.
- Robust regression modeling with cross-validation tuning.
- Clear, interactive dashboards that help stakeholders interpret results quickly.

The project showcases strong skills in data engineering, machine learning, and data storytelling, delivering a complete and modular system suitable for further enhancement and deployment.
