Week 6 – Data Pipeline Testing and Monitoring

Introduction:

As sensor-driven production grows more prevalent in modern manufacturing, real-time anomaly detection is essential for preventing equipment failure, cutting downtime, and guaranteeing product quality. SmartFactory, week 6. AI focuses on developing a strong monitoring pipeline that generates visual warnings to assist operator decision-making, applies an anomaly detection algorithm based on rate-of-change (RoC), and scans simulated streaming data.

Objective:

- Continuously consume Week 5 batches of streaming sensor data.
- Find anomalies almost instantly by applying a rate-of-change thresholding technique.
- Create visual plots with irregular patterns that are legible by humans.
- Maintain visual diagnostics and thorough anomaly reports for future training and auditing.

Methodology:

- **Configuration Configuration:**
  Important variables that can be changed:
  - STREAMED_DATA_PATH: Indicates the location of batch files produced during Week 5.
  - ROLLING_WINDOW: Regulates how much context is kept for detection (by default, 5 records).
  - For every sensor metric, ROC_THRESHOLD specifies acceptable rate-of-change limitations, such as temperature > 15°C, rpm > 500, vibration > 1.0, etc.

  These cutoff points serve as dynamic baselines for identifying possible issues.
- **Logic for Detecting Anomalies:**
  Put into practice in detect_anomalies(df):
  - Sequential scanning of a rolling window of data is done.
  - The script calculates the absolute difference between each key metric's current and prior values for every new row.

    - An anomaly is recorded with the following information if the rate of change above its corresponding threshold:
        - Timestamp
        - Sensor Values
        - The anomaly's explanation (for example, "Δ 20.1 > 15.0" for temperature)

      For later usage, all of these highlighted records are compiled into an anomalies list.

- **Pipeline for Real-Time Monitoring:**
  Put into practice in monitor_batches():
    - Reads every file from the Week 5 output folder that starts with "stream_batch" in an iterative manner.
    - Timestamps are parsed once each batch is transformed into a Pandas DataFrame.
    - Keeps track of the latest ROLLING_WINDOW + 1 rows in a rolling window buffer (rolling_df).
    - The detecting function is called for each new batch.
        - Calls the detecting function.
        - If anomalies are found:
            - Saves them as JSON
            - Calls the visualization function

  Anomaly_summary.json is a compilation of the overall number of anomalies found.
- **Visual Diagnostics:**
  Managed by visualize_anomalies(batch_name, df, anomalies):
    - Creates line charts showing each important metric over time.
    - Red scatter dots are used to draw attention to anomalies over the typical signal.
    - Matplotlib is used to save one plot per metric to Week6/AnomalyPlots/.

  These images are perfect for dashboards in upcoming weeks and provide operator-facing proof for root cause analysis.

Results & Observations:

- **Effective Sudden Sensor Spike Detection:**
    - The algorithm reliably detects abrupt changes in temperature, vibration, RPM, voltage, and pressure.
    - Thanks to established thresholds, even little changes in RoC are detected.

- **Scalable Window-Based Processing:**
    - The rolling window adds temporal continuity across sensor data and avoids overfitting to batch-specific patterns.
    - Makes certain that even minor variations between batches may be identified.
- **Clear Output:**
    - Among the anomaly reports are:
        - Precise detecting timestamp.
        - Breakdown of abnormalities by metric.
        - Snapshot of the sensor data at the anomalous moment.
- **Visual Outputs Enhance Interpretability:**
    - Plots make a clear distinction between spikes and typical trends.
    - Give operators prompt indications so they can look at it.

Summary:

A fully operational real-time monitoring system that reads data as though from live sensors, contextualizes it, and generates actionable insights is put into place in week six. The anomaly detection method based on RoC is:

- Easy: No ML training is needed.
- Quick: Uses batch-on-arrival functionality.
- Interpretable: Provides thorough explanations that are readable by humans.

Conclusion:

A real-time anomaly detection framework that operates on top of simulated IoT sensor feeds is a milestone that SmartFactory.AI has reached at the conclusion of Week 6. Predictive maintenance and downtime mitigation are made possible by ensuring that the factory is not just linked but also aware of issues as they arise. This infrastructure will now be used in Week 7 to streamline processes and strategically rearrange activities in response to identified problems.

URL To Week 6 – Data Pipeline Testing and Monitoring.ipynb