IoT Sensor Data Analysis Challenge

Problem Statement

Objective: As an AI intern working on an IoT project, your task is to analyze temperature data from a sensor in a controlled environment to identify recurring temporal patterns using machine learning. The dataset is provided as a CSV file with columns `device_id`, `ts` (Unix timestamp in milliseconds), and `temperature`. You will model the expected temperature behavior, detect deviations from these patterns, and implement alerting rules for significant anomalies.

Tasks

1. Data Loading and Preprocessing

- Load the CSV file into a data structure.
- Convert the `ts` column from Unix timestamps (milliseconds) to a human-readable format: `YYYY-MM-DD HH:MM:SS`.
- Handle missing or invalid data appropriately (e.g., imputation, removal, or flagging).

2. Pattern Analysis with Machine Learning

- Use a machine learning model to capture periodic temperature patterns (e.g., cycles over minutes or hours).
- Extract insights from the model, such as:
  - Cycle period (e.g., time between peaks).
  - Amplitude (e.g., range of temperature fluctuations).
- Report model performance using metrics like Mean Squared Error (MSE) and R^2 score.

3. Deviation Detection

- Identify significant deviations by comparing actual and predicted temperatures (e.g., `|actual - predicted| > 2°C`).
- Add an `is_deviation` column to the dataset:
  - `1` for records with deviations.
  - `0` otherwise.

4. Visualization

- Create a time-series plot showing:
  - Actual temperatures.
  - Predicted temperatures.
  - Markers for deviations.

- Plot the cyclic pattern (e.g., predicted temperature over one or more cycles).
- Generate a histogram of prediction errors, highlighting the deviation threshold (e.g., 2°C).
- Save all plots as PNG files:
    - `timeseries.png`
    - `cycle_pattern.png`
    - `error_histogram.png`

5. Alerting Rules

- Define rules for generating alerts based on deviations (e.g., threshold-based or consecutive deviations).
- Summarize alerts in a report or table, including:
    - Timestamp.
    - Actual temperature.
    - Predicted temperature.
    - Reason for the alert (e.g., deviation magnitude).

6. Output and Report

- Save a CSV file for each temperature dataset with the following columns:
    - `device_id`
    - Reformatted `ts` (YYYY-MM-DD HH:MM:SS)
    - `temperature`
    - `predicted_temperature`
    - `is_deviation`
- Provide a summary report (e.g., in a markdown cell, document, or code comments) that includes:
    - Total number of records processed.
    - Key patterns identified (e.g., cycle period, temperature range).
    - Number of deviations detected and alerts generated.
    - Model performance metrics (e.g., MSE, R^2).
    - Comparison of patterns between the two temperature datasets (e.g., differences in cycle period or amplitude).

7. Bonus: Motor Current Analysis (Optional)

- Apply your machine learning method to the motor current dataset (CSV with columns `device_id`, `ts`, `motor_current` in amperes).
- In a markdown cell or document, discuss:
    - How well the method performed on the current data compared to the temperature data.
    - Any challenges encountered (e.g., noisier data, different periodicity).
    - Differences in patterns between temperature and current data.

Requirements

- **Language**: Python is preferred, but other languages are acceptable.
  - **Python**: Submit as a Jupyter notebook.
  - **Other languages**: Submit a well-documented script with a separate report document.
- **Input**:
  - Temperature CSV 1 (temperature-data-device-1.csv)
  - Temperature CSV 2 (temperature-data-device-2.csv)
  - Motor current CSV (motor-data-device-3.csv)
  - All CSVs contain: `device_id`, `ts` (Unix timestamp in milliseconds), and either `temperature` (°C) or `motor_current` (amperes).
- **Output**:
  - CSV files for each dataset with the specified columns.
  - Image files for visualizations.
  - A Jupyter notebook (Python) or script (other languages) with clear documentation and implementation.
- The use of AI tools is permitted, but you must disclose in your report how they were used and for what purposes (e.g., code generation, analysis).