Smart Factory Alert Agent Development

Hsieh Yu Hsuan
School of Computing
National University of Singapore

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Smart Factory Alert Agent Development

I. Task Description

The objective of this project is to develop a real-time anomaly detection pipeline for monitoring industrial sensor data, including temperature, pressure, and vibration readings.

The system integrates:

- A rule-based detector that flags readings exceeding predefined thresholds.
- A machine learning–based detector using an unsupervised model to identify abnormal patterns.
- A unified alerting system that merges both detection sources for better accuracy and interpretability.
- Data visualization for sensors and alert occurrences.

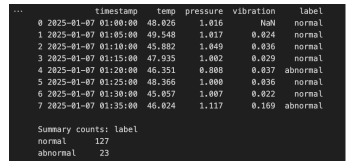
The final solution provides:

1) Real-time anomaly detection.
2) Severity scoring.
3) Visualization with threshold overlays.
4) Text-based alerts for interpretability.

II. Workflow

The end-to-end workflow consists of the following steps, as shown in Figure 6:

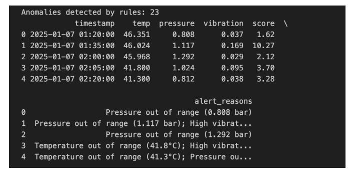
1) **Data Collection & Preprocessing**
   - This step is simulated by generating 150 rows of data with outliers (anomalies) and missing cells.
   - Sensor readings are collected with timestamps.
   - Missing values are handled using forward fill and imputation.
   - Sensor values are standardized for model input.
   - An example of generated data is shown in Figure 1
2) **Rule-based Anomaly Detection**
   - Abnormal thresholds are set for each sensor based on the instruction.
   - Anomalies are flagged when readings fall outside the thresholds.
   - A rule score is calculated based on deviation from threshold.
   - Reasoning text based on rules are provided.
   - An example of rule-based anomaly detection is shown in Figure 2
3) **Machine Learning–based Anomaly Detection**
   - We utilized a supervised machine learning approach with a **Random Forest Classifier**.
   - To understand why the model classified certain readings as abnormal, we used **SHAP (SHapley Additive exPlanations)** [1].
   - An example of ML-based anomaly detection and reasoning is shown in Figure 3
4) **Alert Merging & Score Alignment**
   - Anomalies from both sources are merged on timestamps.



```
...              timestamp    temp  pressure  vibration     label
    0 2025-01-07 01:00:00  48.026     1.016        NaN    normal
    1 2025-01-07 01:05:00  49.548     1.017      0.024    normal
    2 2025-01-07 01:10:00  45.882     1.049      0.036    normal
    3 2025-01-07 01:15:00  47.935     1.002      0.029    normal
    4 2025-01-07 01:20:00  46.351     0.808      0.037  abnormal
    5 2025-01-07 01:25:00  48.366     1.000      0.036    normal
    6 2025-01-07 01:30:00  45.057     1.007      0.022    normal
    7 2025-01-07 01:35:00  46.024     1.117      0.169  abnormal

    Summary counts: label
    normal      127
    abnormal     23
```

Fig. 1. Generated data



```
Anomalies detected by rules: 23
            timestamp    temp  pressure  vibration  score  \
0 2025-01-07 01:20:00  46.351     0.808      0.037   1.62
1 2025-01-07 01:35:00  46.024     1.117      0.169  10.27
2 2025-01-07 02:00:00  45.968     1.292      0.029   2.12
3 2025-01-07 02:05:00  41.800     1.024      0.095   3.70
4 2025-01-07 02:20:00  41.300     0.812      0.038   3.28

                                    alert_reasons
0                    Pressure out of range (0.808 bar)
1  Pressure out of range (1.117 bar); High vibrat...
2                    Pressure out of range (1.292 bar)
3  Temperature out of range (41.8°C); High vibrat...
4  Temperature out of range (41.3°C); Pressure ou...
```

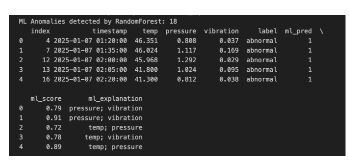
Fig. 2. Rule-based anomaly detection

   - Scores are normalized to comparable ranges (0-1).
   - Analogy reasons for rule-based and ML-based are shown in terminal.
   - An example of rule-based anomaly detection is shown as Figure 4
5) **Visualization & Reporting**
   - Anomalies are displayed on a time-series plot.
   - Thresholds are shown as horizontal lines for reference.
   - Alerts are printed with details and suggestions.
   - An example of temperature is shown as Figure 5

III. Data Descriptions

For the purpose of developing and testing the anomaly detection system, we generated a dummy dataset simulating sensor readings from industrial equipment, as illustrated in Table I. The dataset consists of 150 sequential records with the following fields:

- **timestamp:** The recorded time for each observation, provided at one-minute or five-minute intervals.
- **temp:** Temperature readings in degrees Celsius. Normal values range between 45 and 50°C, while abnormal readings fall outside this range (greater than 52°C or less than 43°C).



```
ML Anomalies detected by RandomForest: 18
   index           timestamp    temp  pressure  vibration     label  ml_pred  \
0      4 2025-01-07 01:20:00  46.351     0.808      0.037  abnormal        1
1      7 2025-01-07 01:35:00  46.024     1.117      0.169  abnormal        1
2     12 2025-01-07 02:00:00  45.968     1.292      0.029  abnormal        1
3     13 2025-01-07 02:05:00  41.800     1.024      0.095  abnormal        1
4     16 2025-01-07 02:20:00  41.300     0.812      0.038  abnormal        1

   ml_score       ml_explanation
0      0.79  pressure; vibration
1      0.91  pressure; vibration
2      0.72       temp; pressure
3      0.78      temp; vibration
4      0.89       temp; pressure
```

Fig. 3. ML-based anomaly detection

Fig. 4. First 5 rows of combined alerts

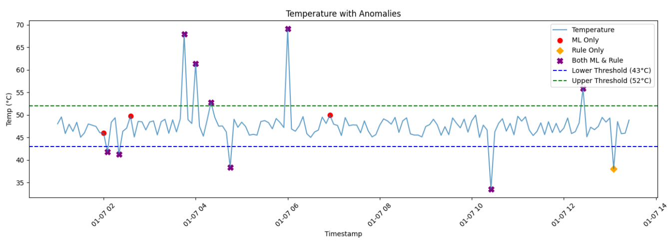


Fig. 5. Visualization of temperature data

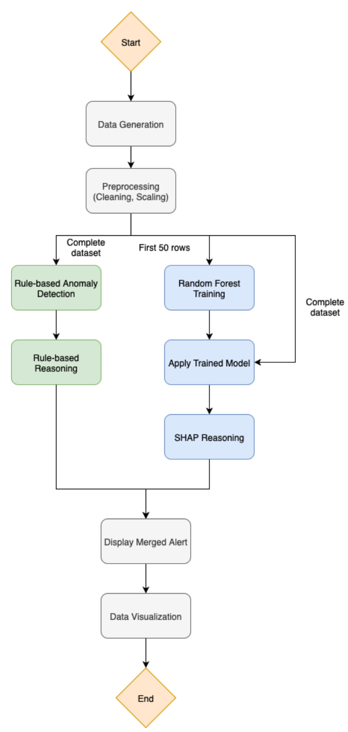


Fig. 6. Project workflow

- **pressure:** Pressure measurements. Normal values range from 1.00 to 1.05, whereas abnormal readings exceed 1.08 or are below 0.97.
- **vibration:** Vibration intensity. Normal readings are between 0.02 and 0.04, and abnormal values exceed 0.07.
- **label:** Status label indicating whether the observation is normal or abnormal, as determined either by the sensor system or the AOI (Automated Optical Inspection) machine.

timestamp	temp	pressure	vibration	label
2024-06-03 19:05:00	48.2	1.01	0.03	normal
2024-06-03 19:06:00	47.9	1.01	0.03	normal
2024-06-03 19:07:00	50.3	1.06	0.06	abnormal
2024-06-03 19:08:00	46.5	1.02	0.02	normal
2024-06-03 19:09:00	52.5	1.10	0.08	abnormal

TABLE I. SAMPLE RECORDS FROM THE DUMMY DATASET.

IV. MODEL DETAILS

In this project, we employ a hybrid anomaly detection system combining both rule-based and machine learning-based approaches. This section focuses on the machine learning component.

A. Model Selection

We selected **Random Forest Classifier** for supervised anomaly detection due to its robustness to small datasets, ability to handle non-linear relationships, and interpretability. Given that the sensor data contains labeled instances of normal and abnormal states, a supervised learning approach is appropriate.

B. Training Procedure

- **Training Data:** Only the first 50 rows of the dataset are used for training. This simulates a realistic scenario where limited labeled anomalies are available.
- **Features:** The input features include standardized sensor measurements: temperature, pressure, and vibration.
- **Target Variable:** The label column is converted to a binary numeric format (0 = normal, 1 = abnormal) for training the classifier.
- **Scaling:** All feature values are standardized using z-score normalization to ensure comparable ranges across features.
- **Model Hyperparameters:**
  - Number of estimators: 100
  - Maximum tree depth: None (fully grown trees)
  - Class weight: “balanced” to account for class imbalance due to rare anomalies
  - Random seed: 50 for reproducibility

C. Prediction

After training, the Random Forest model predicts on the entire dataset to identify anomalies. Two outputs are generated for each sample:

1) **Prediction Label:** Binary classification indicating normal or abnormal.
2) **Probability Score:** The predicted probability of being abnormal, which is used as the ML-based anomaly score.

D. Interpretability with SHAP

To provide explanations for each anomaly, we integrate **SHAP (SHapley Additive exPlanations)**:

- SHAP computes the contribution of each feature to the prediction probability for class 1 (abnormal).
- For each detected anomaly, we generate a textual explanation highlighting which sensor measurements contributed most to the classification.
- Contributions with negligible effect (SHAP value $<$ 0.01) are filtered out to focus on only significant factors.

E. Integration with Rule-Based Detection

The ML-based scores are combined with rule-based scores to produce a unified alert table:

- Rule-based scores are normalized to a similar scale as ML-based probability scores to enable comparison.
- Alerts are categorized as detected by Rule, ML, or Both.
- Explanations from SHAP are displayed alongside rule-based reasons to provide both data-driven and domain-driven insights.

This combination allows the system to leverage both domain knowledge and supervised learning to detect and explain anomalies effectively.

V. AI Usage In The Work

AI tools were actively leveraged throughout the development process to enhance productivity, generate ideas, and streamline implementation. Specifically:

- Idea Generation and Discussion: ChatGPT [2] was used to brainstorm and discuss implementation strategies, helping to refine the workflow and identify potential challenges early in the process.
- Code Development and Debugging: GitHub Copilot [3], integrated in VSCode, was used to generate initial code skeletons based on the discussed workflow, as well as to assist in debugging and iterating on code efficiently.
- Document Preparation and LaTeX Support: AI features in Overleaf were utilized to debug LaTeX syntax issues, format equations, and generate tables, reducing manual effort and ensuring correctness.
- Report Refinement: ChatGPT was also used to improve the clarity, structure, and readability of this report, providing suggestions for phrasing and organization.
- Additional Assistance: AI tools were also consulted for quick clarification of technical concepts, best practices in coding, and suggestions for optimizing algorithms, which helped improve both the quality and speed of development.

References

[1] Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 4765–4774). Curran Associates, Inc. https://arxiv.org/abs/1705.07874

[2] OpenAI. (2025). *ChatGPT* (GPT-4) [Large language model]. OpenAI. https://chat.openai.com

[3] GitHub. (2025). *GitHub Copilot* [AI-powered code completion]. GitHub. https://github.com/features/copilot