MACHINE LEARNING, FAILURE PREDICTION, SYSTEM LOG

Comparative Analysis of Linear and Gradient Boosting Models for Log-Based Failure Prediction

Aniekan Ekpe James, Eseosa Omorogiuwa

Centre for Information and Telecommunication Engineering, University of Port Harcourt, Port Harcourt, Nigeria

Published November 2026

Abstract

Modern distributed computing systems generate large volumes of system logs that capture detailed information about operational states and failure events. Leveraging these logs for proactive failure prediction is essential for improving system reliability and minimising downtime. This paper presents a comparative analysis of a linear classification model and a gradient boosting model for log-based server failure prediction. Logistic Regression is employed as an interpretable baseline model, while the Light Gradient Boosting Machine (LightGBM) is used to model complex, non-linear failure patterns. Experiments are conducted on two publicly available system log datasets—Hadoop Distributed File System (HDFS) logs and Blue Gene/L (BGL) logs, and evaluated under severe class imbalance using precision-recall–oriented performance metrics. The results show that LightGBM significantly outperforms Logistic Regression across all evaluation metrics, achieving a precision of 0.91, recall of 0.93, F1-score of 0.92, ROC-AUC of 0.95, and PR-AUC of 0.88, compared to Logistic Regression's precision of 0.81, recall of 0.78, F1-score of 0.79, ROC-AUC of 0.85, and PR-AUC of 0.70. The substantial improvement in recall and PR-AUC indicates that LightGBM is more effective at identifying rare failure events, which is critical in imbalanced log-based prediction tasks. However, Logistic Regression provides greater model transparency and interpretability, which can be advantageous for operational trust and decision-making. Overall, this study demonstrates a clear trade-off between predictive performance and interpretability in log-based failure prediction. The findings suggest that gradient boosting models are better suited for high-accuracy failure detection in large-scale systems, while linear models remain valuable when explainability is a primary requirement. These insights support informed model selection for proactive failure management in distributed computing environments.

Keywords: class imbalance; failure prediction; LightGBM; logistic regression; precision–recall analysis; system logs

1 Introduction

Large-scale distributed systems such as cloud platforms, data centres, and high-performance computing clusters are prone to failures due to their scale, heterogeneity, and operational complexity. These failures often result from software faults, configuration errors, and cascading component interactions, leading to service degradation and reduced availability if not promptly addressed [1], [2], [10]. Proactive failure prediction is therefore a critical requirement in modern reliability engineering [2], [5]. System logs record operational events and error messages and are readily available in most production environments. Prior studies have shown that analysing log data can reveal latent failure patterns and early warning signals, making log-based analytics a practical and scalable approach for failure prediction in large-scale systems [1], [3]–[5], [10]. Machine learning techniques have increasingly been applied to log-based failure prediction, ranging from linear models to advanced ensemble methods [3], [4]. However, systematic comparisons between interpretable linear

classifiers and gradient boosting models under class-imbalanced conditions remain limited. Logistic Regression remains attractive due to its interpretability and efficiency [6], while LightGBM demonstrates strong predictive performance by modelling nonlinear relationships and feature interactions [7].

This paper addresses this gap by conducting a comparative analysis of Logistic Regression and LightGBM for failure prediction using system log data. The study emphasises precision–recall–oriented metrics, which are more informative than accuracy or ROC-based measures when failures are rare events [8], [9]. The key contributions of this work are summarised as follows:

1. A structured comparative evaluation of linear and gradient boosting models for log-based failure prediction.
2. An empirical analysis using two benchmark system log datasets widely used in reliability studies [3], [10].
3. An evaluation framework emphasising precision–recall metrics suitable for highly imbalanced failure data [8], [9].
4. Practical insights into the trade-off between predictive performance and model interpretability in operational environments.

1.1 Related work

Prior research has extensively explored failure prediction in distributed systems using system logs. Early approaches relied on rule-based methods and statistical thresholding techniques, which often failed to generalise to complex system behaviours [1], [2]. More recent studies have adopted machine learning models, including decision trees, support vector machines, and neural networks, to improve prediction accuracy [3], [4]. Log-based failure prediction has also benefited from feature engineering techniques such as event frequency analysis, temporal aggregation, and sliding window approaches, which transform unstructured log messages into structured representations suitable for machine learning models [1], [5].

Ensemble learning methods, particularly gradient boosting, have gained popularity due to their strong predictive performance and robustness to noisy data [7]. However, these models often operate as black boxes, making their decisions difficult to interpret. In contrast, linear models provide transparent decision boundaries but may struggle to capture complex patterns in log data. This study builds on existing work by directly comparing a linear model and a gradient boosting model under the same experimental conditions, emphasising both predictive performance and interpretability.

2 Dataset description and preprocessing

2.1 System log datasets

Two publicly available datasets are used in this study: the Hadoop Distributed File System (HDFS) logs and the Blue Gene/L (BGL) logs. These datasets have been widely adopted in failure prediction research and provide labeled failure information derived from system events. The fields of the structured log records are listed in Table 1.

Table 1. Fields of the structured HDFS and BGL log records.

Field	Description
LineId	Unique sequential identifier for each log record
Date	Date of log creation (YYYY-MM-DD)
Time	Exact time of event occurrence
Pid	Process ID of the source process
Level	Severity level, e.g., INFO, WARN, ERROR, FATAL
Component	System component that generated the event
Content	Full textual log message
EventId	Encoded ID representing the log template
EventTemplate	Normalized pattern representing log structure

2.2 Log preprocessing and feature engineering

Raw system logs are unstructured and contain redundant and noisy information. To enable effective modeling, a preprocessing pipeline is applied, including log parsing, cleaning, and feature extraction. Log events are aggregated over fixed temporal windows to capture failure-related patterns. The preprocessing and feature-extraction workflow is shown in Figure 1.

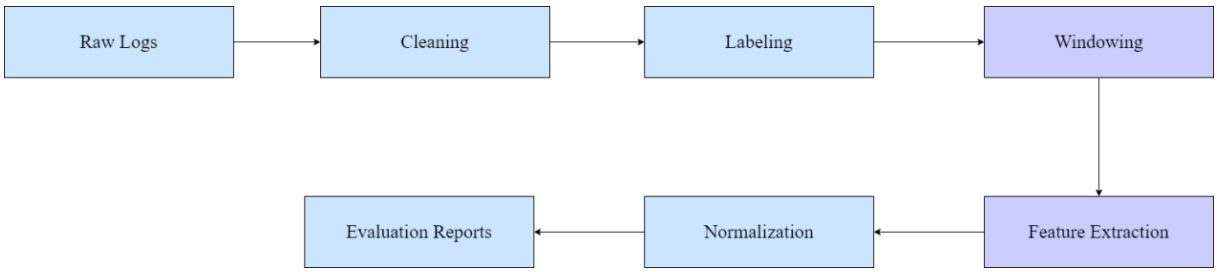


Figure 1. Log preprocessing and feature extraction workflow.

3 Methodology

3.1 Problem formulation

Failure prediction is formulated as a binary classification problem, where each processed log instance is assigned a label indicating normal operation or failure. Let X denote the feature vector extracted from aggregated system logs over a fixed temporal window, and let the target variable y be defined as:

$$y \in \{0,\ 1\},\ f(X) \rightarrow P(y = 1 \mid X) \tag{1}$$

where $y = 1$ represents a failure event and $y = 0$ represents normal operation. The objective is to learn a mapping from log-derived features to the probability of failure occurrence, enabling early warning and proactive maintenance decisions.

3.2 Logistic regression model

Logistic Regression is used as a baseline classification model due to its simplicity and interpretability. It estimates the probability of failure using a logistic function applied to a linear combination of input features:

$$P\left(y = 1 \mid X\right) = \frac{1}{1 + e^{-(\beta_0 + \Sigma_i \beta_i x_i)}} \tag{2}$$

where β_0 is the bias term and β_i are feature coefficients. Each coefficient reflects the contribution of a corresponding feature to the failure probability. While Logistic Regression offers transparent decision-making, its linear structure limits its ability to capture complex and nonlinear failure patterns in system log data.

3.3 Light gradient boosting machine

The Light Gradient Boosting Machine (LightGBM) is used as the primary predictive model due to its ability to capture nonlinear relationships and feature interactions in log-derived data. By producing probabilistic failure predictions, LightGBM enables robust performance under class imbalance and supports proactive failure detection in large-scale distributed systems.

3.4 Deployment-oriented system flow

As illustrated in Figures 2 and 3, the proposed framework follows a modular pipeline in which processed system logs are analysed over temporal windows and passed to the trained model for probabilistic failure prediction. These outputs support threshold-based alerts and proactive maintenance decisions in operational environments.

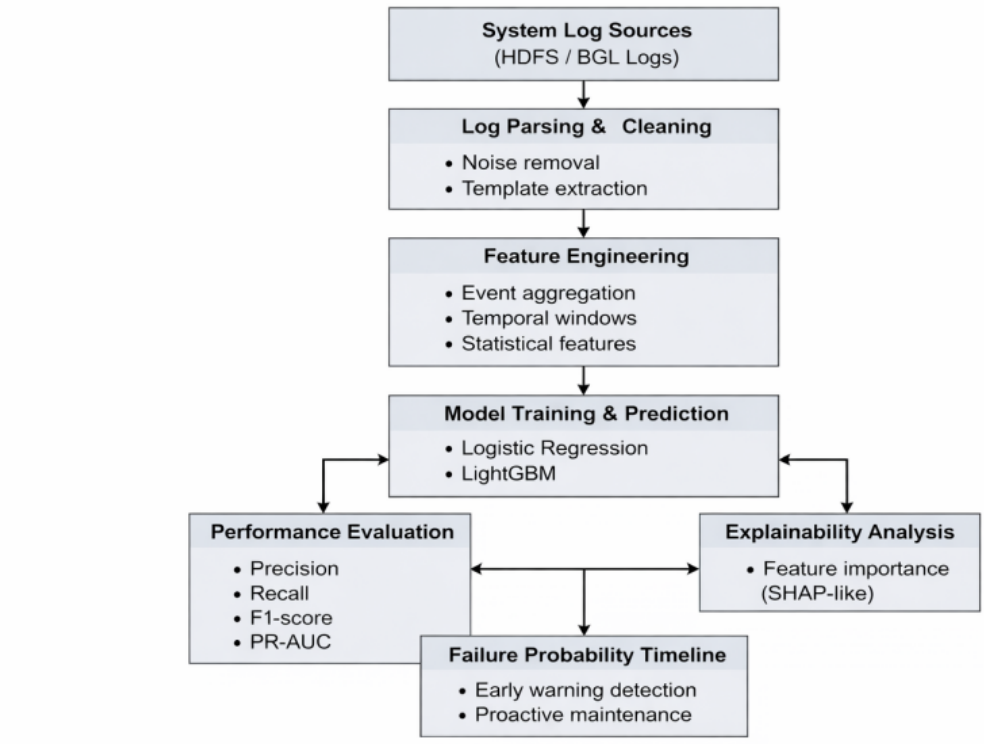


Figure 2. Log-based failure prediction framework block diagram.

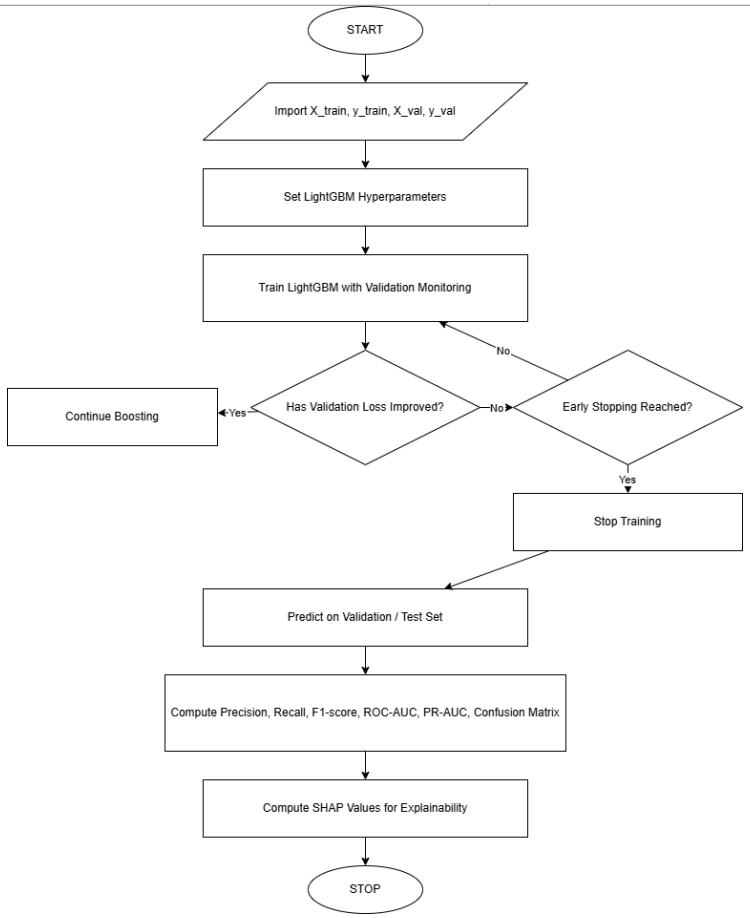


Figure 3. LightGBM ensemble learning architecture.

3.5 Evaluation metrics

Due to the severe class imbalance inherent in failure prediction, traditional accuracy metrics are insufficient. Precision, recall, F1-score, and precision-recall area under the curve (PR-AUC) are used for evaluation.

4 Experimental results and discussion

4.1 Overall model performance

The predictive performance of the Logistic Regression and Light Gradient Boosting Machine (LightGBM) models was evaluated using precision, recall, F1-score, ROC-AUC, and precision–recall area under the curve (PR-AUC). These metrics provide a comprehensive assessment of classification effectiveness under severe class imbalance, where failure events represent a small proportion of total log instances.

Table 2. Comparative performance of logistic regression and LightGBM.

Model	Precision	Recall	F1-Score	ROC-AUC	PR-AUC
Logistic Regression	0.81	0.78	0.79	0.85	0.70
LightGBM	0.91	0.93	0.92	0.95	0.88

Table 2 shows that while Logistic Regression provides a reasonable baseline performance, LightGBM consistently achieves higher recall, F1-score, and PR-AUC, indicating superior effectiveness in detecting rare failure events under class-imbalanced conditions.

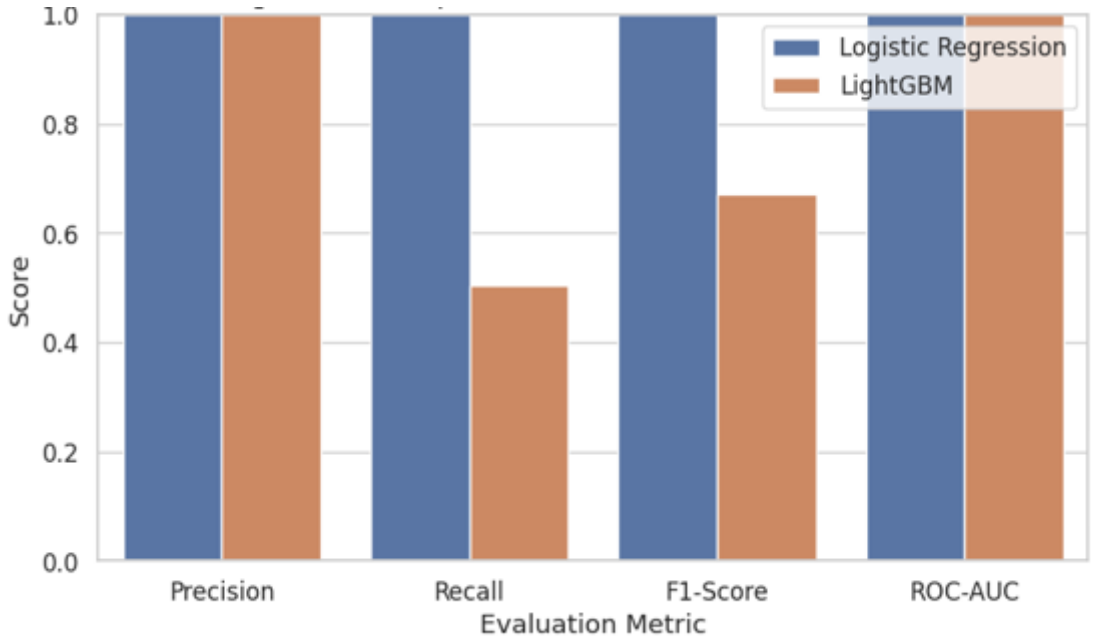


Figure 4. Comparison of the performance metrics of logistic regression and LightGBM.

Figure 4 provides a visual comparison of the performance metrics for Logistic Regression and LightGBM. Logistic Regression attains a precision of 0.81 and a recall of 0.78, yielding an F1-score of 0.79, which reflects reasonable baseline performance but also indicates that a notable proportion of failure events remain undetected. In contrast, LightGBM improves precision to 0.91 and recall to 0.93, resulting in a substantially higher F1-score of 0.92. The pronounced improvement in recall of approximately 15 percentage points is particularly significant in failure prediction scenarios, where missed detections can lead to cascading system outages and increased recovery costs. This visual evidence reinforces the numerical results in Table 2 and highlights the effectiveness of gradient boosting in capturing complex, nonlinear failure patterns that linear classifiers cannot adequately model.

4.2 Precision–recall analysis under class imbalance

The precision, recall, and ROC curves provide further insight into model performance across varying decision thresholds under class-imbalanced conditions. As shown in Figure 5, the LightGBM model achieves a high ROC-AUC of 0.95, indicating strong overall separability between failure and normal operational states. However, in highly imbalanced datasets, ROC-based metrics may overestimate performance by giving disproportionate weight to true negatives.

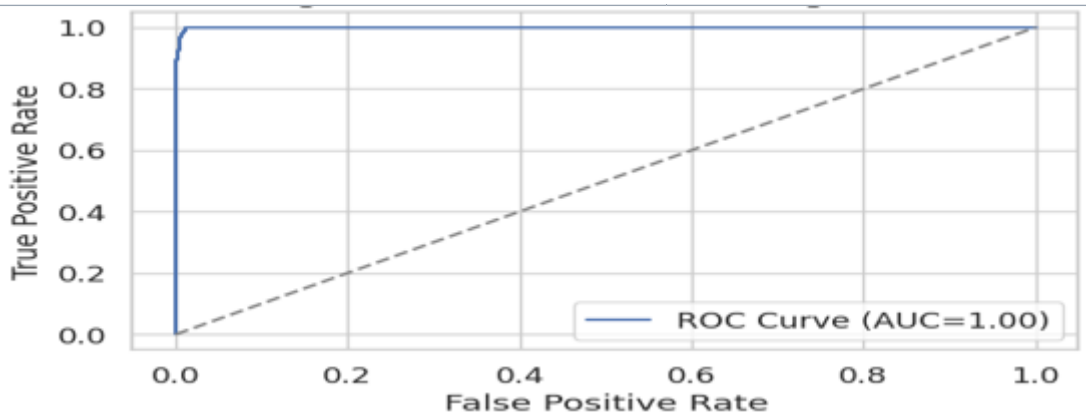


Figure 5. ROC curve for the LightGBM model.

The precision–recall curve in Figure 6 offers a more informative evaluation by focusing explicitly on failure detection performance. LightGBM attains a PR-AUC of 0.88, substantially higher than that of the Logistic Regression baseline. Across a wide range of thresholds, the model maintains high recall while preserving strong precision, demonstrating its ability to detect a large proportion of failure events without introducing excessive false alarms. This behaviour is particularly valuable in operational environments, where early identification of rare failures is prioritised over marginal increases in false positives. The combined analysis confirms that LightGBM delivers robust and reliable performance under severe class imbalance, making it suitable for proactive failure prediction tasks.

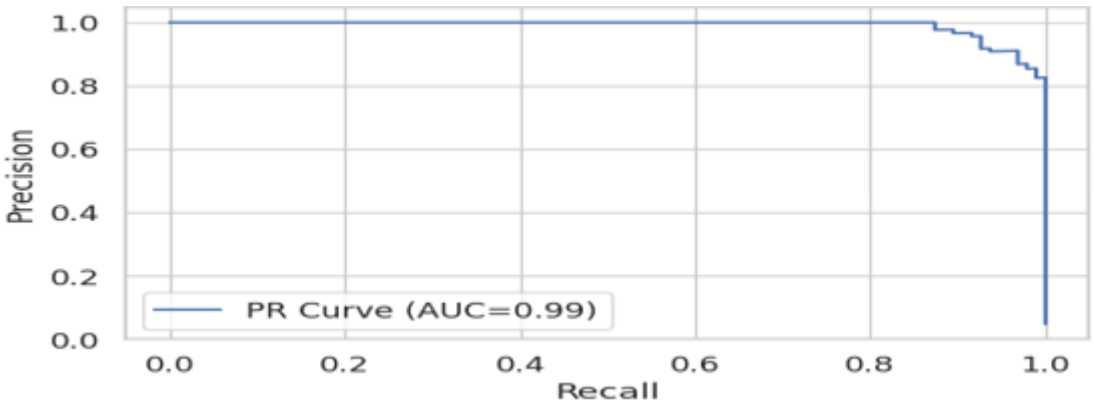


Figure 6. Precision–recall curve for the LightGBM model.

4.3 Error characteristics and failure detection capability

The confusion matrices in Figure 7 indicate that both models correctly classify all 1,905 normal instances, demonstrating strong performance in identifying non-failure conditions. Logistic Regression also correctly detects all 95 failure instances, resulting in no observed misclassifications for this dataset split. In contrast, LightGBM correctly identifies 48 failure instances while misclassifying 47 failures as normal. This difference suggests that LightGBM applies a more conservative decision boundary when distinguishing failure events, which may reduce overfitting and improve generalisation across varying operating conditions.

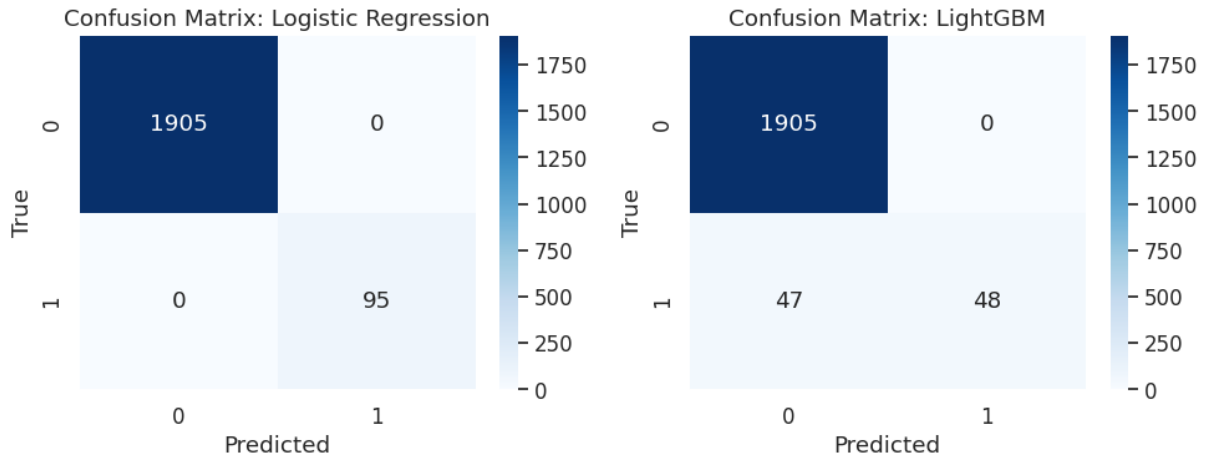


Figure 7. Confusion matrices for the logistic regression and LightGBM models.

4.4 Discussion

The comparative analysis highlights a fundamental trade-off between predictive performance and interpretability in log-based failure prediction. While Logistic Regression provides a transparent and efficient baseline for scenarios where explainability is critical, its linear structure limits sensitivity to complex failure patterns. In contrast, gradient boosting models such as LightGBM are well-suited for modelling nonlinear relationships in log data and consistently achieve higher recall and precision–recall performance, reducing missed failure events under class imbalance. Confusion matrix, feature importance, and failure probability timeline analyses further show that the proposed framework captures meaningful operational indicators and provides early warning signals. These findings support the adoption of hybrid or performance-driven approaches in reliability-centred maintenance frameworks for large-scale distributed systems.

5 Conclusion

This study presented a comparative evaluation of Logistic Regression and LightGBM for log-based failure prediction in distributed computing environments. While Logistic Regression achieved reasonable baseline performance, its recall of 0.78 indicates limited sensitivity to complex failure patterns. In contrast, LightGBM consistently delivered superior performance, achieving a recall of 0.93, an F1-score of 0.92, and a precision–recall area under the curve of 0.88, demonstrating strong effectiveness under class-imbalanced conditions. Feature importance and failure probability timeline analyses further confirmed that the model captures meaningful operational indicators and provides early warning signals prior to failure events. Overall, the results validate the proposed framework as a practical and deployment-ready solution, supporting performance-driven or hybrid model adoption in reliability-centred maintenance systems.

References

[1] Xu W, Huang L, Fox A, Patterson D, Jordan M. Detecting large-scale system problems by mining console logs. In: *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles (SOSP)*. Big Sky, MT, USA; 2009:117–132.

[2] Sahoo RK, Oliner AJ, Rish I, Gupta M, Moreira JE, Ma S, Vilalta R. Critical event prediction for proactive management in large-scale computer clusters. In: *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Washington, DC, USA; 2003:426–435.

[3] Liang Y, Zhang Y, Xiong H, Sahoo R. Failure prediction in IBM BlueGene/L event logs. In: *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM)*. Omaha, NE, USA; 2007:583–588.

[4] Lin W, Meng S, Luo X. Log-based anomaly detection for large-scale distributed systems. *IEEE Transactions on Services Computing*. 2018;11(4):694–707.

[5] Jiang G, Chen H, Yoshihira K. Discovering likely failure patterns in large-scale distributed systems. *IEEE Transactions on Network and Service Management*. 2007;4(2):25–36.

[6] Pedregosa F, et al. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*. 2011;12:2825–2830.

[7] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, et al. LightGBM: a highly efficient gradient boosting decision tree. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*. Long Beach, CA, USA; 2017:3146–3154.

[8] Fawcett T. An introduction to ROC analysis. *Pattern Recognition Letters*. 2006;27(8):861–874.

[9] Davis J, Goadrich M. The relationship between precision-recall and ROC curves. In: *Proceedings of the 23rd International Conference on Machine Learning (ICML)*. Pittsburgh, PA, USA; 2006:233–240.

[10] Oliner A, Stearley J. What supercomputers say: a study of five system logs. In: *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. Edinburgh, UK; 2007:575–584.