IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 28, Issue 4, Ser. 2 (July. – August. 2026), PP 70-77
www.iosrjournals.org

Evaluation of Machine Learning Models for Real-Time Fault Detection in Industrial Production Lines

Jiali He[1], Chen Yuan*

[1](Center for Applied Mathematics of Guangxi, Yulin Normal University , Yulin, Guangxi 537000, P.R. China,

Abstract
Intelligent fault alarm technology plays a critical role in enhancing the efficiency of automated production lines by enabling real-time monitoring and early fault detection. In this study, we analyzed one year of operational data from ten industrial production lines to systematically investigate fault characteristics. Key features during fault occurrences were extracted to train and evaluate multiple machine learning models. Comparative results demonstrated that the Bagged Trees model achieved superior performance, with higher classification accuracy and F1 scores than competing approaches. Subsequent validation on an independent test set confirms that the model maintains robust fault detection capability without overfitting. These findings indicate its strong potential for real-world deployment in industrial settings.

Keywords*: Automatic alarm prediction, Machine learning, Accuracy, F1 scores*

Date of Submission: 09-08-2026 Date of Acceptance: 21-08-2026

I. INTRODUCTION

A. Background of the Study

Current automation control technology has reached the mechatronic stage, completing the transition from semi-intelligent to fully intelligent systems [1]. These intelligent systems rely on machine-based information processing and autonomous decision-making, where automatic real-time alarm technology serves as a core component of intelligent manufacturing. By implementing real-time monitoring and rapid response mechanisms, this technology significantly enhances both the intelligence and automation levels of production processes. It enables precise identification of production anomalies, thereby improving production line intelligence and driving the progressive development of smart manufacturing. As technological advancements accelerate and application fields expand, the industrial significance of real-time alarm systems continues to grow [2].

Specifically, automatic production lines are equipped with integrated sensors, monitoring devices, and other advanced technologies to collect real-time data on the production environment and equipment operating status. When abnormalities in equipment are detected by the system, the real-time alarm mechanism is immediately triggered, assisting workers in quickly taking corrective measures, thereby minimizing downtime and improving production efficiency. More importantly, this technology incorporates predictive maintenance capabilities, allowing warnings to be issued before equipment faults occur, enabling preventive maintenance. This not only extends the service life of the equipment but also significantly enhances its reliability and maintenance efficiency [3].

In automated production environments, flexible personnel allocation strategies prove particularly critical for adapting to dynamic production tasks and workflow variations. Scientific workforce deployment ensures stable production line operation, maximizes productivity, and enhances market responsiveness to fluctuating demands [4]. Importantly, both overstaffing and understaffing are recognized as cost-increasing factors: the former leads to human resource waste, while the latter may cause production bottlenecks or inefficiencies. Through optimized personnel allocation, enterprises can simultaneously reduce unnecessary labor costs and avoid productivity-related economic losses, achieving dual improvements in economic and social benefits [5-6].

B. Research Significance

Accurate fault identification allows enterprises to reduce unnecessary equipment inspections, downtime maintenance, and associated costs, while also avoiding the potential risks of fault escalation. This reduces operational costs and maintenance expenses for production lines. Automated fault detection systems facilitate early equipment maintenance, minimizing damage caused by failures, extending equipment lifespans, and

improving resource utilization. Intelligent manufacturing is a key direction for industrial development, with fault detection and diagnosis serving as core issues. Research in this area provides technical support for the digital and intelligent transformation of the manufacturing industry, enhancing the overall competitiveness of the sector and driving the development of intelligent manufacturing [7].

C. Research Objective

This study is conducted based on one year of operational data collected from 10 production lines in a Chinese factory [8]. The fault data characteristics of various devices in the automated production lines are systematically analyzed, and a corresponding fault warning model is established.

To gain a deeper understanding of the fault characteristics of each device, the data associated with faults in each device is first filtered out, and the relationships with other variables linked to the faults are systematically analyzed. As shown in Figure 1, by integrating the production line operation flowchart and the fault descriptions of each device, the variables associated with faults in each device and their intrinsic connections are further clarified. Figure 1 shows the functions of each step in the production line and the faults that may occur. For example, the red rectangle represents transports materials from the Material Pushing Device to the Material Detection Device. The error code it generates is 1001. The content enclosed by the orange rectangle consists of Main conveyor belt, Material Pushing Device, and Material Detection Device. They cause a malfunction in material detection, with fault code 2001.

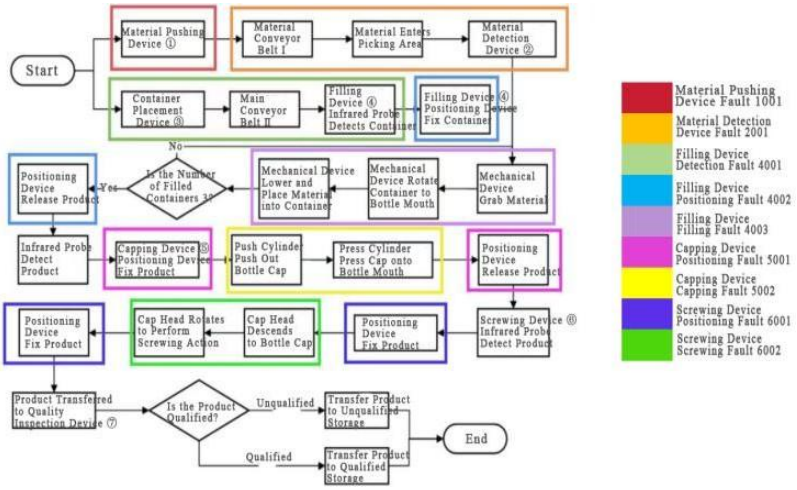

Figure 1 Production line operation process and faults generated by various devices [8]

II. DATA DESCRIPTION AND PREPROCESSING

A. Data Characterization

The annual operational records of the 10 production lines in this factory comprise a total of 75,289,095 data entries, covering 28 operational parameters and 9 fault categories. The case distribution and data overview for each production line are detailed in Figure 2. Table 1 shows the 28 feature types and functions of the production line dataset. As shown in the Table 1 (Row 2, Column 5), the 'Material push status' feature captures the logistic feeding operation status, which is stored as categorical data.

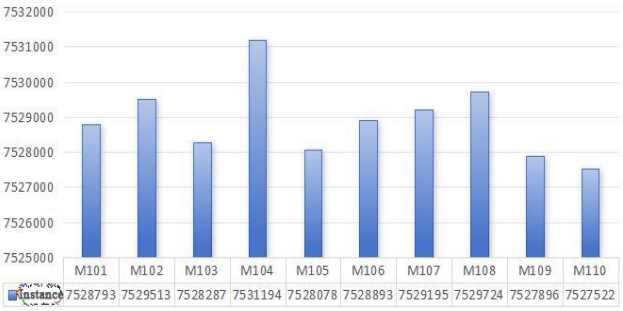

Figure 2: Instances statistics of 10 production lines

Table no 1: Feature and data type summary for the production line dataset [8].

	28 Feature Descriptions			
Feature	(1) Date	(2) Time	(3) Production line number	(4) Material push status
Type	Temporal	Temporal	Categorical	Categorical
Feature	(5) Material retraction status	(6) Number of material push notifications	(7) Number of materials to be grabbed	(8) Number of containers placed
Type	Categorical	Categorical	Categorical	Categorical
Feature	(9) Number of container detections	(10) Number of filling inspections	(11) Fixed state of filling locator	(12) Release status of filling locator
Type	Categorical	Categorical	Categorical	Categorical
Feature	(13) Number of material grabs	(14) Fill in rotation number	(15) Decrease in filling quantity	(16) Filling quantity
Type	Categorical	Categorical	Categorical	Categorical
Feature	(17) Number of stamped detections	(18) Cover the positioning number	(19) Number of push covers	(20) Cover the decrease number
Type	Categorical	Categorical	Categorical	Categorical
Feature	(21) Number of stamps added	(22) Number of twist cap inspections	(23) Screw cap positioning number	(24) Lower number of screw caps
Type	Categorical	Categorical	Categorical	Categorical
Feature	(25) Twist cap rotation number	(26) Number of screw caps	(27) Qualified quantity	(28) Unqualified number
Type	Categorical	Categorical	Categorical	Categorical

B. Data Cleaning

In the process of data analysis, it is essential to ensure that no duplicate cases exist for key variables. For production line operation data, the device operation records for each production line within a single day should contain only unique timestamps. Records with identical timestamps are identified as duplicate cases. In this study, dates and timestamps are used as matching criteria to identify and remove duplicate samples, resulting in a high-quality, cleaned dataset.

C. Exploratory Data Analysis

The 10 production lines included in the study each consist of 9 stages where faults may occur, corresponding to Fault 1001, Fault 2001, Fault 4001, Fault 4002, Fault 4003, Fault 5001, Fault 5002, Fault 6001, and Fault 6002. The statistical results of the fault rates for these 10 production lines are presented in Figure 3. From Figure 3, it can be observed that Fault 1001 occurs most frequently in production line M101, Fault 5002 occurs most frequently in M102, Faults 4002 and 6001 occur relatively frequently in M103, Fault 4001 occurs most frequently in M104, Fault 5002 occurs most frequently in M105, Fault 1001 occurs most frequently in M106, M107, and M108, while Fault 5002 occurs most frequently in M109 and M110.

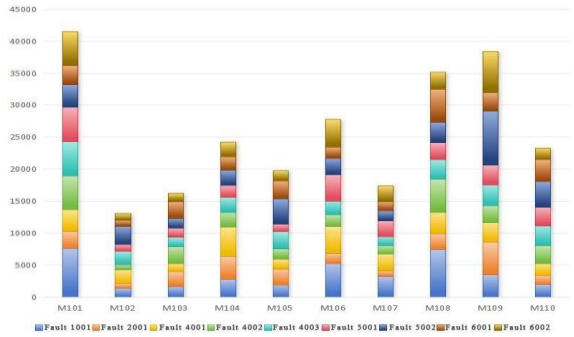

Figure 3 Frequency distribution of 10 production line faults

Subsequently, statistical analysis was performed on the total fault rates across all 10 production lines, with results summarized in Table 2. The analysis revealed that production lines M101 and M109 exhibited relatively high fault rates, whereas M102 demonstrated the lowest rate. During the one-year observation period, the 10 production lines collectively recorded 256,692 fault instances, yielding an average fault rate of 0.34%.

Table no 2: Fault rates of 10 production lines.

Production line	Number of instances	Fault frequency	Fault rate
M101	7528793	41479	0.55%
M102	7529513	13049	0.17%
M103	7528287	16212	0.22%
M104	7531194	24206	0.32%
M105	7528078	19766	0.26%
M106	7528893	27722	0.37%
M107	7529195	17370	0.23%
M108	7529724	35170	0.47%
M109	7527896	38412	0.51%
M110	7527522	23306	0.31%
Total	75289095	256692	0.34%

III. METHODOLOGY

Following the completion of data preprocessing, the machine learning model training phase is initiated for fault detection implementation. Several industrially validated effective models are subsequently presented for this specific task.

A. Algorithms Selection

A.1 Classification and Regression Trees

Classification and Regression Trees (CART) is a versatile decision tree algorithm that supports both classification and regression tasks [9]. It recursively splits the data by selecting features and thresholds that maximize purity (e.g., minimizing Gini impurity for classification or mean squared error for regression). The process continues until a stopping criterion is met, and pruning is often applied to prevent overfitting. The Gini Index for Classification is defined as follows:

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^{\,2}$$

In decision tree classification tasks, p_i represents the proportion (probability) of samples belonging to the i-th class in the current data node . It is formally defined as:

$$p_i = \frac{\text{Number of samples of class } i}{\text{Total number of samples in the node}}$$

A.2 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning model used for classification and regression [10]. The key idea is to find the optimal hyperplane that maximizes the margin between different classes. For non-linear classification, SVM uses kernel functions to map data into a higher-dimensional space where separation is possible. A common kernel is the radial basis function :

$$K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2)$$

where γ controls the influence of each training sample.SVM solves the following constrained optimization problem:

$$\min \frac{1}{2}\|w\|^2 \quad \text{subject to } y_i(wx_i + b) \geq 1 \quad \forall i$$

where $y_i \in \{-1,1\}$ are the class labels.

A.3 Naive Bayes

Naive Bayes (NB) is a probabilistic classification algorithm based on Bayes' Theorem, known for its simplicity and efficiency, especially in high-dimensional problems like text classification [11]. Its core formula is:

$$P(y \mid X) = \frac{P(y)\Pi_{i=1}^{n} P(x_i \mid y)}{P(X)}$$

Here, y is the class label, $X = (x_1, \cdots, x_n)$ represents feature values. The algorithm predicts the class with the highest $P(y|X)$.

The classifier offer two principal advantages that make them particularly valuable for specific machine

learning applications: computational efficiency and implementation simplicity. This algorithm demonstrates exceptional performance in high-dimensional data classification tasks and scenarios with limited training samples, as it operates through direct probability estimation rather than iterative optimization.

A.3 k-Nearest Neighbors

k-Nearest Neighbors (KNN) is a simple yet powerful instance-based supervised learning algorithm used for both classification and regression [12]. It operates on the principle that similar data points tend to have similar outcomes.

The algorithm steps are as follows:

1. Training Data: Labeled dataset;
2. Test Point: Unlabeled query point;
3. Distance Metric: Typically Euclidean distance or Manhattan distance;
4. Hyperparameter *k*: Number of nearest neighbors to consider;

KNN is widely used in recommendation systems, anomaly detection and medical diagnosis.

A.4 Subspace Discrimination Method

The Subspace Discrimination Method (SDM) is a classification technique that operates by projecting data into lower-dimensional subspaces and performing discrimination (classification) in these subspaces [13]. It is particularly useful for high-dimensional data, such as images, text, or gene expression data, where traditional methods may suffer from the curse of dimensionality. The key steps of an algorithm are:

1. Subspace Learning (Compute projection matrix using unsupervised PCA or supervised SVM for discriminant analysis);
2. Dimensionality Reduction (Project high-dimensional data into the discriminant subspace);
3. Subspace Modeling (Train classifiers like SVM/K-NN in the low-dimensional space);
4. Inference Prediction (New samples are first projected then classified).

A.5 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is probabilistic classification methods that model class-conditional data distributions [14]. It works well for multi-class problems and provide both classification and dimensionality reduction capabilities. The decision rule of LDA is

$$\delta_k(x) = x^T \sum^{-1} \mu_k - \frac{1}{2} \mu_k^T \sum^{-1} \mu_k + \log P(y = k) .$$

LDA assumes equal covariance matrices across classes, creates linear decision boundaries by maximizing class separation through dimensionality reduction, and serves for both classification and supervised feature projection.

A.6 Bagged Trees

Bagged Trees (BT) is an ensemble learning method that improves the stability and accuracy of decision trees by combining multiple independently trained trees [15]. The key idea is to reduce variance and prevent overfitting through majority voting for classification.

The Bagging Trees proceeds with the following specific steps:

1. Assume the number of sample sets is *k*;
2. Generate *k* datasets of size n, where each bootstrap sample set has the same size as the original dataset;
3. Train classifiers on the *k* sample sets;
4. Determine the final classification result through voting.

Bagged Trees are widely used in Random Forests, which further improve performance by randomizing feature selection during splitting.

B. Numerical Experiment

In this study, multiple machine learning algorithms, including CART, SVM, NB, KNN, SDM, LDA and BT models are employed for training. The training steps are as follows:

Step 1: Input Cleaned dataset (features x and class labels y).

Step 2: Divide the data into 80% training set and 20% test set, maintaining class balance.

Step 3: Choose baseline models (e.g., CART, KNN). Train the model using the training set (X_train, y_train).
Step 4: Model Evaluation. Predict class labels (y_pred) using the test set (X_test) and Calculate evaluation metrics: Accuracy, Precision & Recall, F1-Score. Output a classification report .
Step 5: Cross-Validation. Perform 5-fold cross-validation on the training set to compute average performance and ensure model stability.
Step 6: Parameter Tuning. Define a parameter search space. Use Grid Search or Random Search to find optimal parameters.
Step 7: Output Final Model. The best-performing classifier and test Set Performance.

The evaluation metrics used for the trained model in these papers are classification accuracy and F1 score [16]. The parameters for data sampling are shown in Table 3, and the model evaluation results are displayed in Table 4. As shown in Table 3, the second row lists the feature set used for model training, specifically covering the 4th to 28th feature dimensions, while the third row details the training sample composition, including n fault samples and 10,000 randomly selected non-fault samples, forming the complete training dataset.

Table no 3: Model Parameter Settings.

Parameter Name	Parameter values
Selected features	From the 4th feature to the 28th feature
Number of faults	Number (The number of faults)
Number of training sets	Number＋10000 (Number of instances without faults)

Table no 4: Model Evaluation [16].

Name	Description
F1 score	The harmonic mean of F1 model accuracy and recall
Classification accuracy	The precision of classification

All numerical experiments were conducted on a workstation equipped with an Intel Core i7-11800H processor (2.3 GHz base frequency) and 32 GB RAM, running Windows 10 Professional. The algorithms were implemented in MATLAB R2021a, with the Machine Learning Toolbox utilized for model development and evaluation.

IV. RESULT AND ANALYSIS

A. Performance Comparison

A subset of fault data was extracted from the historical operational records of the production lines and paired with normal data points, with the number of normal samples exceeding the total fault instances. The combined data was randomly shuffled to form the training set, while the remaining original unextracted data were retained as the test set. Seven distinct machine learning models - CART, SVM, NB, KNN, SDM, LDA and BT were constructed using these datasets.

The model development process involved training, testing, hyperparameter tuning, and predictive performance evaluation, enabling comprehensive model comparison and selection. To account for the randomness in data extraction and ensure prediction stability, we conducted over 10 iterations of randomized data matching and machine learning experiments. Finally, we statistically aggregated the predictive evaluation results for each model and computed the mean F1-scores and classification accuracies, as presented in Tables 5-6.

Table no 5: F1 score evaluation results for each model.

Model	F1 score	Ranking
CART	0.9574	5
SVM	0.9379	6
NB	0.8727	7
KNN	0.9937	2
SDM	0.9640	4
LDA	0.9779	3
BT	0.9986	1

Table no 6: Classification accuracy evaluation results for each model.

Model	Classification accuracy	Ranking
CART	0.9253	5
SVM	0.8895	6
NB	0.7900	7
KNN	0.9897	2
SDM	0.9379	4
LDA	0.9639	3
BT	0.9976	1

As evidenced in Tables 5-6, all evaluated models with the exception of NB achieved classification accuracy and F1-scores exceeding 90% on the training set. These results demonstrate exceptional training efficacy, with satisfactory evaluation metrics indicating high classification precision and stability.
The models' performance ranked as follows (in descending order): BT> KNN > LDA > SDM > CART > SVM> NB. Notably, the Bagged Trees algorithm exhibited statistically superior performance compared to other methods.
Given these findings, we focus subsequent discussion on potential overfitting risks in the BT model. Its stability and generalizability were further validated through predictive performance testing on the production line test set.

B. Robustness Validation

To further validate whether overfitting occurs, the remaining data from each production line are used as the test set to evaluate the model's generalization capability [17]. In the previous section, the BT model was identified as the best performer. Therefore, in this section, the BT model is examined for potential overfitting, and its performance on the test set is compared with that on the training set.
The test datasets were input into the trained BT model, and the results are shown in Tables 7(a,b).

Table no 7 (a) : Fault prediction and evaluation indicators for each device with BT.

Production line	M101		M102		M103		M104		M105	
Number	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy
1001	0.9566	0.9484	0.9338	0.941	0.9189	0.92	0.9445	0.9322	0.9366	0.9351
2001	0.9606	0.9622	0.91	0.9229	0.9465	0.9588	0.9221	0.9338	0.9459	0.9544
4001	0.9755	0.9808	0.9605	0.9444	0.942	0.9395	0.934	0.945	0.9652	0.9668
4002	0.9766	0.9802	0.9621	0.9549	0.9786	0.9818	0.9283	0.9121	0.9744	0.9552
4003	0.956	0.9661	0.939	0.9316	0.9543	0.97	0.9324	0.9248	0.9561	0.952
5001	0.9371	0.9357	0.9278	0.9261	0.9381	0.9366	0.9245	0.9452	0.9485	0.9446
5002	0.9876	0.9709	0.936	0.9424	0.9515	0.9404	0.9674	0.955	0.956	0.9763
6001	0.9323	0.9349	0.9366	0.9245	0.9756	0.9722	0.9744	0.972	0.9153	0.924
6002	0.9774	0.9762	0.978	0.9764	0.9708	0.9765	0.9332	0.9633	0.9284	0.9365
Average	0.9622	0.9617	0.9426	0.9405	0.9529	0.9551	0.9401	0.9426	0.9474	0.9494

Table no 7 (b): Fault prediction and evaluation indicators for each device with BT.

Production line	M106		M107		M108		M109		M110	
Number	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy
1001	0.9469	0.952	0.9878	0.9898	0.9561	0.9643	0.9565	0.975	0.9568	0.9577
2001	0.966	0.9595	0.997	0.9881	0.9774	0.9651	0.9544	0.962	0.9642	0.9653
4001	0.9677	0.9596	0.9754	0.9765	0.9846	0.9854	0.945	0.9463	0.9475	0.9566
4002	0.9457	0.9438	0.9823	0.9875	0.974	0.9711	0.9577	0.9475	0.9245	0.931
4003	0.9654	0.9424	0.9747	0.9657	0.9752	0.9663	0.9658	0.9678	0.9333	0.936
5001	0.9325	0.9375	0.9845	0.9745	0.971	0.9618	0.935	0.9466	0.9455	0.9327
5002	0.9655	0.9533	0.9632	0.9751	0.9688	0.9642	0.9322	0.9441	0.9323	0.9442
6001	0.9124	0.921	0.9659	0.955	0.9515	0.9424	0.9581	0.9563	0.9557	0.9562
6002	0.945	0.9339	0.9722	0.9734	0.9553	0.9477	0.9632	0.9688	0.9445	0.9313
Average	0.9497	0.9448	0.9781	0.9762	0.9682	0.9631	0.952	0.9572	0.9449	0.9457

By analyzing Tables 7(a,b)，it can be clearly seen that BT demonstrates exceptional performance in fault detection tasks. On the independent test sets across all production lines (M101-M110), this model demonstrated excellent fault detection performance, achieving an average accuracy and F1 score above 0.94. This indicates that the trained model not only maintains highly accurate fault identification but also exhibits outstanding generalization capabilities across different production lines.

This demonstrates that the proposed model not only delivers consistently stable fault detection rates exceeding 94%, thereby validating its high reliability and strong generalization capability, but also enables direct reduction of over 90% of manual re-inspection costs, satisfying all core requirements for industrial deployment.

V. CONCLUSIONS

This paper addresses the problem of fault detection and identification in industrial production lines through comprehensive characterization of process-generated data. Multiple visualization techniques—including bar charts and histograms—were employed to intuitively present the multidimensional characteristics of the production data. These methods not only enable thorough dataset comprehension but also facilitate multiperspective data analysis.

Following systematic analysis of production line equipment fault patterns, we developed a fault alarm model and conducted comparative performance evaluation of seven machine learning algorithms: CART, SVM, NB, KNN, SDA, LDA and BT. Through rigorous comparison of classification accuracy and F1-scores, the BT model demonstrated superior performance, achieving 94% classification accuracy. Consequently, we selected BT as the optimal solution for production line fault prediction to enhance operational efficiency. The model's robust performance across varying production conditions establishes a new benchmark for fault detection systems in smart manufacturing, particularly in handling imbalanced fault data distributions where traditional methods typically underperform.

ACKNOWLEDGEMENT

This work was supported by the Guangxi Natural Science Foundation under Grant No.2026GXNSFHA00640129 and the College Students' Innovation and Entrepreneurship Program under Grant No.X2026106060018

References

[1] Lv Y, Yang X, Li Y, et al. Fault detection and diagnosis of marine diesel engines: A systematic review[J]. Ocean Engineering, 2024, 294: 116798.

[2] Chaleshtori A E, Aghaie A. A novel bearing fault diagnosis approach using the Gaussian mixture model and the weighted principal component analysis[J]. Reliability Engineering & System Safety, 2024, 242: 109720.

[3] Mercorelli P. Recent advances in intelligent algorithms for fault detection and diagnosis[J]. Sensors, 2024, 24(8): 2656.

[4] Sudhakar P K, Muthucumaraswamy R, Selvaraju P, et al. A Regularized Mixed Integer Linear Programming framework with penalties for integrated workforce, subcontracting and production optimization[J]. JIMO, 2026, 22(4): 2014-2043.

[5] Kulkarni A J. Optimization Methods in Staff Scheduling and Allocation[M]//Optimization Methods in Manufacturing Processes: A Machine-Generated Literature Overview. Singapore: Springer Nature Singapore, 2025: 433-488.

[6] Olivieri G, Mangini A M, Fanti M P. Optimizing personnel allocation: An integer linear programming problem for enhanced workplace efficiency[J]. Computers & Industrial Engineering, 2025, 212: 111724.

[7] Zhu Y, Zhao S, Zhang Y, et al. A review of statistical-based fault detection and diagnosis with probabilistic models[J]. Symmetry, 2024, 16(4): 455.

[8] "Competition question," Tipdm Big Data Platform, 2023. [Online]. Available: https://www.tipdm.org:10010/#/competition/1734744522337984512/question. Accessed: Jul. 20, 2024.

[9] Yu P, Chen Y, Xu C, et al. Binary split categorical feature with mean absolute error criteria in CART[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2026, 40(33): 27961-27968.

[10] Qiu J, Xie J. Multi-attribute intuitionistic fuzzy twin support vector machine based on data distribution[J]. Information Sciences, 2026: 123203.

[11] Mostefaoui K, Mostefaoui S A M, Mekroussi S, et al. Fault detection in industrial battery production using naive Bayes networks[J]. 1, 2026, 11(2): 97.

[12] Hassanat A B, Alkasasbeh A A, Alkafaween E, et al. On the Optimality of k= $\sqrt{n}$ in k-Nearest Neighbor Classification: Sub-Optimality Rates, Dimension-Aware Selection, and Hassanat Distance Comparison[J]. Mathematics, 2026, 14(15): 2707.

[13] Almarzooqi A, Arab M G, Omar M, et al. Benchmarking Conventional Machine Learning Models for Dynamic Soil Property Prediction. Buildings 2025, 15, 4188[EB/OL]. critical review(2021)

[14] Liu J, Feng M, Xiu X, et al. Towards robust and sparse linear discriminant analysis for image classification[J]. Pattern Recognition, 2024, 153: 110512.

[15] Soleimani A, Amin S H, Baki F, et al. Forecasting weekly sales using lag-based feature engineering and a bagged tree ensemble[J]. Machine Learning with Applications, 2026: 100922.

[16] Zhang Z, Wang J, Wen F, et al. Large multimodal models evaluation: a survey[J]. Science China Information Sciences, 2025, 68(12): 221301.

[17] Halabaku E, Bytyçi E. Overfitting in Machine Learning: A Comparative Analysis of Decision Trees and Random Forests[J]. Intelligent Automation & Soft Computing, 2024, 39(6): 1-20.