APEM journal

Advances in Production Engineering & Management
Volume 20 | Number 4 | December 2025 | pp 458–474
https://doi.org/10.14743/apem2025.4.552

ISSN 1854-6250
Journal home: apem-journal.org
Original scientific paper

A multi-objective feature selection and self-paced ensemble framework for semiconductor defect detection

Zheng, H.[a], Gao, X.[a,*], Yang, X.[a], Jing, G.[a], Yang, M.[a], Liu, Y.[a]

[a]School of Mechanical Engineering, Xi'an University of Technology, Xi'an, P.R. China

ABSTRACT

In semiconductor manufacturing, defect detection is commonly performed using high-dimensional process data. These data often exhibit class imbalance and class overlap, which create challenges for achieving reliable classification performance. To address these issues, this study proposes a multi-objective feature selection and self-paced ensemble (MOFS-SPE) framework. The framework employs a multi-objective evolutionary algorithm based on decomposition (MOEA/D) for feature selection. In this process, the area under the precision–recall curve (AUPRC) and the *R*-value are used as objective functions to identify feature subsets that are highly relevant to quality outcomes. In addition, the framework integrates the self-paced ensemble (SPE) with tree-based classifiers to handle imbalanced and overlapping data. Experiments conducted on a real semiconductor manufacturing dataset (SECOM dataset) demonstrate the effectiveness of the proposed approach. Compared with using the full feature set, the selected features increase the area under the receiver operating characteristic curve (AUROC) from 0.685 to 0.770 and the AUPRC from 0.932 to 0.972. When applying the SPE framework, the specificity of the decision tree model improves from 0.048 to 0.667, thereby enhancing the reliability of identifying defective products. Overall, the proposed framework provides a useful reference for intelligent quality inspection in semiconductor production environments.

ARTICLE INFO

Keywords:
Semiconductor manufacturing;
Defect detection;
Quality inspection;
Class imbalance;
Class overlap;
Multi-objective feature selection;
Self-paced ensemble;
Machine learning

**Corresponding author:*
xaut66gxq@163.com
(Gao, X.)

Article history:
Received 27 August 2025
Revised 30 November 2025
Accepted 3 December 2025

1. Introduction

Product quality is a critical factor for the sustainable development of manufacturing enterprises [1]. As a result, achieving "zero defects" has become one of the primary goals of lean production managemesnt. In high-end manufacturing sectors such as semiconductor fabrication, the production process typically involves multiple tightly coupled stages [2], including photolithography, etching, and deposition. The complexity of both the production systems and process flows makes product quality highly sensitive to subtle variations [3]; even minor deviations at any stage can lead to performance degradation or complete failure of the final product. If defective products reach the market, they not only damage the company's reputation but also incur substantial costs due to customer complaints and warranty claims [4]. Therefore, establishing efficient and reliable quality management and inspection mechanisms has become a central concern for manufacturers in the semiconductor industry.

To address the above challenges, quality inspection has become one of the most critical means of ensuring product reliability in semiconductor manufacturing [5]. In a typical wafer fabrication

process, inspection stages are implemented after key processing steps to assess product quality using various methods. Common inspection approaches include manual visual inspection and specialized detection equipment. However, manual inspection is inefficient, labor-intensive, and highly subjective, with results often influenced by the operator's experience, leading to false detections or missed defects [6]. Although dedicated inspection equipment offers higher precision, it is expensive to procure and maintain, and often requires complex integration and system modifications when process changes occur [7]. Therefore, how to ensure high detection accuracy while reducing inspection costs and improving efficiency has become an urgent and pressing issue for semiconductor manufacturers.

With the rapid advancement of data-driven technologies, intelligent quality inspection methods based on manufacturing data have emerged as a promising solution in the semiconductor industry. The semiconductor fabrication process involves multiple precision processing stages, each generating large volumes of high-dimensional sensor data and process parameters. When product quality anomalies occur, the manufacturing data often deviates from the normal data range [7]. Machine learning techniques can analyse complex relationships in high-dimensional process data and establish mapping models between process parameters and product quality [8]. These techniques can classify product quality based on manufacturing data [9]. By leveraging machine learning, manufacturers can achieve cost-effective and efficient quality inspection without the need for additional inspection equipment or personnel.

In practical industrial production, issues of class imbalance and class overlap pose significant challenges to quality inspection. The data for quality-compliant products is far more abundant than that for defective products. Consequently, the raw manufacturing data is typically highly imbalanced. Several studies have addressed the characteristics of class imbalance in quality inspection [10]. However, class imbalance is not the only challenge in quality inspection. More importantly, class overlap significantly impacts the accuracy of machine learning models [11]. When class overlap occurs in manufacturing data, data points of different quality grades overlap in the feature space, making it difficult for the model to accurately distinguish between quality-compliant and non-compliant products. This considerably reduces the model's discrimination ability and increases the likelihood of misclassification.

This study aims to develop a data-driven approach for automated quality inspection in semiconductor manufacturing, with the goal of ensuring high detection accuracy while effectively reducing inspection costs. Although existing studies have explored data-driven product quality inspection methods, most of them focus on classification accuracy, neglecting the impact of class overlap and class imbalance on the performance of quality inspection models. When data is imbalanced, using classification accuracy alone as an evaluation standard for quality inspection models often results in a higher false positive rate [12]. This means that defective products might be erroneously classified as quality-compliant, leading to higher misclassification costs.

To address the challenges of quality inspection under conditions of class overlap and class imbalance in semiconductor manufacturing, this study proposes a data-driven intelligent inspection framework. The framework aims to improve classification accuracy in the presence of class imbalance and class overlap through feature selection and model development. Firstly, a multi-objective optimization algorithm based on the MOEA/D, using AUPRC and *R*-value [13] as objective functions, is employed to select a subset of features highly related to quality from high-dimensional manufacturing data. Secondly, the SPE [14] framework is introduced, incorporating various tree-based classifiers as base estimators to enhance quality inspection accuracy under imbalanced and overlapping class conditions. The main innovations and contributions of this study are summarized as follows:

- An intelligent quality inspection framework tailored for semiconductor manufacturing is proposed, encompassing key stages such as manufacturing data preprocessing, feature selection, classification model construction, and performance evaluation.
- A multi-objective feature selection algorithm based on MOEA/D is designed, using AUPRC and *R*-value as objective functions, to select a subset of process features from high-dimensional data that can reduce class overlap and improve classification accuracy.

- A quality inspection model was constructed by integrating the SPE framework with tree-based classifiers, demonstrating strong adaptability and generalization capability for defect detection in complex manufacturing scenarios.
- The proposed method is validated using a semiconductor manufacturing dataset, demonstrating the superiority of the feature selection algorithm and classification model in addressing class imbalance and class overlap issues.

2. Related works

Quality inspection methods can be divided into traditional methods and machine learning-based methods. Furthermore, the importance of data imbalance and class overlap issues in quality inspection research has been increasingly recognized. Therefore, this section will review representative studies from three perspectives: traditional quality inspection methods, machine learning-based quality inspection methods, and the current research status on data imbalance and class overlap issues.

2.1 Traditional quality inspection methods

Traditional quality inspection primarily relies on statistical analysis methods. Zhou *et al*. [15] proposed a quality defect diagnosis method for multi-stage manufacturing processes based on a generalized matrix regression model, which demonstrated higher diagnostic accuracy through real-world case studies. Sun *et al*. [16] proposed a distributed kernel principal component regression (DKPCR) model for quality inspection and diagnosis, addressing quality-related process inspection at the factory level. Although these traditional methods exhibit high diagnostic accuracy and effectiveness in specific applications, they face significant challenges in handling high-dimensional data.

2.2 Machine learning-based quality inspection

With the development of technology, the manufacturing industry has gradually accumulated a large amount of usable manufacturing data, making machine learning-based product quality inspection a research hotspot. Quality inspection techniques that leverage machine learning are commonly categorized into two branches: traditional machine learning and deep learning. The former often encompasses models such as Artificial Neural Networks (ANN), Random Forests, and Support Vector Machines (SVM). Liu *et al*. [17] proposed a classification framework based on Random Forests, providing a solution for quality inspection in the lithium-ion battery manufacturing process by effectively quantifying the importance and correlation of manufacturing features. Saqlain *et al*. [18] introduced a quality inspection model based on ensemble learning by voting, integrating four machine learning models: Logistic Regression, Random Forests, Gradient Boosting Machines, and Artificial Neural Networks to improve the accuracy of the quality inspection model. Mlinarič *et al*. [19] proposed a decision tree–based quality inspection approach that utilizes feature importance for preliminary feature selection, followed by the application of Decision Trees, Random Forests, Bagging, and Gradient Boosting classifiers to enhance the accuracy of electric motor classification.

With the success of deep learning technologies in image processing and text processing, researchers have begun to apply these methods to product quality inspection. Azamfar *et al*. [20] propose a domain-adaptive quality diagnosis method based on deep learning, validated using actual semiconductor manufacturing datasets. Stavropoulos *et al*. [21] detect weld seam defects using feature extraction combined with machine learning methods and make intelligent decisions with Markov models. Chen *et al*. [22] present a quality diagnosis model based on stochastic deep Koopman, designed for real-time detection of quality defects in multi-stage production systems. Despite significant advancements in machine learning-based methods for product quality inspection, current research rarely considers the coupled impact of class imbalance and class overlap issues.

2.3 Research on data imbalance and class overlap issues

In the process of product quality inspection, datasets often exhibit characteristics of class imbalance and class overlap. These characteristics make quality inspection more complex. Some data balancing algorithms, classification algorithms, and their integrated methods aim to address this issue, such as SmoteBoost [23], BorderlineSMOTE [24], RUSBoosted [25], and BalanceCascade [26]. The SPE [14] framework is an ensemble learning method specifically designed to handle class imbalance and class overlap data classification problems. The SPE framework does not rely on distance calculations between samples, making it applicable to datasets lacking clear distance metrics. Compared to traditional imbalance learning methods, the SPE framework performs better when dealing with noisy data and highly overlapping classes. Additionally, as a general ensemble framework, SPE can be easily adapted to most existing learning methods, such as tree-based machine learning methods.

3. The MOFS-SPE framework in semiconductor manufacturing

The proposed MOFS-SPE framework for intelligent quality inspection in semiconductor manufacturing consists of four main components: data preprocessing, feature selection, model construction, and performance evaluation.

- Data preprocessing: appropriate preprocessing methods are used to process raw data, ensuring data quality and integrity to prevent abnormal data from affecting the accuracy of the quality prediction model.
- Feature selection: a multi-objective feature selection algorithm based on MOEA/D, using AUPRC and *R*-value as objective functions, selects a subset of process features from high-dimensional data to reduce class overlap and improve classification accuracy.
- Quality inspection model construction: during the quality inspection model construction phase, the SPE framework is integrated with tree-based classifiers to achieve product quality classification.
- Quality inspection model evaluation: to evaluate the classification accuracy of the quality inspection model, key performance indicators such as Recall, Specificity, G-Mean, AUROC, and AUPRC are used.

3.1 Data preprocessing

Due to the influence of uncertain factors, manufacturing data obtained from the production site may contain null values, missing values, and outlier samples. Therefore, appropriate data preprocessing methods are required to handle these anomalies. The data preprocessing steps used in this study are briefly described as follows:

- Redundant feature removal: The raw manufacturing data contains single values, null values, and duplicate features, which are considered redundant features. Additionally, features with a high proportion of missing values contribute insufficiently to the model. In this study, features with a missing value ratio greater than 40 % are also considered redundant. Redundant features do not contribute to quality inspection results and need to be removed to ensure data quality.
- Outlier data point removal: The manufacturing environment is complex, and manufacturing data collection and transmission are susceptible to interference, leading to outliers in the raw manufacturing data. To avoid the interference of outliers on quality inspection results, box plots are used to detect outliers, and these outliers are removed.
- Outlier sample removal: In quality inspection, the completeness of samples is crucial. Samples with more than 15 % missing values are considered insufficient in information and may negatively impact the analysis results. Therefore, to ensure data reliability and the accuracy of quality inspection results, these samples are removed.
- Missing value imputation: Missing values are inevitable in manufacturing data collection due to instrument fluctuations or machine failures. The k-nearest neighbor (KNN) algorithm is

flexible, non-parametric, and easy to implement, making it an effective method for missing value imputation. Thus, this study uses the KNN algorithm to fill in missing values.

- Data normalization: The collected features in the product manufacturing process vary in range. To eliminate differences in data dimensions, the max-min method is used for data normalization.

3.2 Feature selection

In semiconductor manufacturing, process data are typically high-dimensional and exhibit significant class imbalance and class overlap, which pose substantial challenges for effective feature selection. To address this issue, this study introduces a multi-objective feature selection method based on the multi-objective evolutionary algorithm based on decomposition (MOEA/D). The proposed method aims to identify a compact and informative subset of features that reduces class overlap and enhances classification performance. The following subsection details the design of the feature selection algorithm, including the objective functions, encoding scheme, population initialization, crossover, and mutation strategies.

Feature selection objective functions

The proposed feature selection algorithm uses AUPRC and *R*-value as the two objective functions. AUPRC can better evaluate the classification performance of the model on imbalanced datasets, especially in recognizing minority class samples. This is particularly important in quality inspection, where it is crucial to identify the few samples that indicate quality anomalies. Additionally, the *R*-value effectively measures the degree of class overlap among features, helping to select a feature subset that reduces class overlap. The first objective function is defined in Eq. 1.

$$f_1 = AUPRC = \int_0^1 P(R)dR \tag{1}$$

where *P(R)* denotes precision at a given recall rate.

According to [13], the calculation methods for the second fitness function are shown in Eqs. 2-4.

$$R_{aug}(D[V]) = \frac{|C_1|R(C_0)+|C_0|R(C_1)}{|C_0|+|C_1|} \tag{2}$$

$$R(C_i) = \frac{1}{|C_i|}\sum_{m=1}^{|C_i|} \lambda\left(\left|kNN\left(P_{i,m}, U - C_i\right)\right| - \theta\right) \tag{3}$$

$$\lambda(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where C_i represents the set of instances belonging to class i, $P_{i,m}$ denotes the m-th instance of class i, U represents the set of all instances, and *kNN(P, S)* denotes the subset of k nearest neighbors of instance P in set S; θ is the threshold used to determine the number of neighbors from different classes that classify an instance as belonging to the overlap region.

Encoding method and population initialization

In this study, a discrete binary encoding method is used to represent the feature selection scheme. The length of each chromosome corresponds to the number of features in the manufacturing data, with each position having a value of 0 or 1. A value of 1 indicates that the feature is selected, while a value of 0 means the feature is not selected.

For population initialization, randomly initialized feature selection schemes may be overly random, potentially leading to poor convergence of the feature selection algorithm. In this research, a SHAP-based population initialization method is adopted. SHAP (SHapley Additive exPlanations) is based on the Shapley value from game theory and provides comprehensive explanations for each feature in machine learning models [27]. Specifically, before population initialization, the Shapley value for each feature is calculated, as shown in Eq. 5. Then, all features are sorted according to their Shapley values, and features are prioritized based on their importance.

$$\phi_i = \sum_{S \subseteq F} \frac{|S|!(|F|-|S|-1)!}{|F|!}\left[f_{S,c(i)}\left(x_{c(i)}\right) - f_S(x_S)\right] \tag{5}$$

where F represents the set of all features; S denotes a subset of features, which is a subset of F; x_s represents the values of the input features in subset S; $x_{c(i)}$ represents the value of feature i; and f denotes the classification model used for quality inspection.

During the population initialization process, the top 30 % of features with high Shapley values are directly selected, while the remaining 70 % of features with low Shapley values are supplemented using random selection. This initialization strategy not only retains features that significantly contribute to model performance but also increases the diversity of the population, thus enhancing convergence speed and overall performance in the early stages of the algorithm. Algorithm 1 illustrates the proposed population initialization method.

Algorithm 1: Population Initialization Based on Shapley Values

Input: Feature Shapley value array ($feature\_shap$), keep ratio ($keep\_ratio$), population size ($n\_samples$)
Output: Initialized population matrix ($samples$)
1: Compute the total number of features: $num\_features$
2: Sort the features in descending order of Shapley values, obtaining the indices: $sorted\_indices$
3: Calculate the number of top features to keep: $num\_keep = num\_features \times keep\_ratio$
4: Select the indices of the top Shapley value features: $keep\_features = sorted\_indices[: num\_keep]$
5: Initialize the population matrix: $samples = zeros(n\_samples, num\_features)$
6: **for** each individual i from 0 to $n\_samples - 1$ **do**
7: Set the positions corresponding to $keep\_features$ in $samples[i]$ to $True$
8: Randomly select $\frac{len(remaining\_indices)}{2}$ features from the remaining lower Shapley value features, and set their positions to $True$
9: **end for**
10: **Return** initialized population $samples$

Crossover operator

Traditional random crossover operators may disrupt important features, which can degrade the quality of the offspring solutions. This method may fail to retain features crucial to model performance, thereby impacting the overall effectiveness of the feature selection algorithm. To overcome this drawback, a directed crossover operation based on feature Shapley values is adopted in this study. Firstly, features are ranked according to their Shapley values, and a retention ratio is set. Then, the top-ranked features are kept unchanged, while the remaining features undergo random crossover. This approach better preserves important features during the crossover process, enhancing the quality and diversity of the offspring. Algorithm 2 illustrates the proposed crossover operator.

Algorithm 2: Crossover Operator Based on Shapley Values

Input: Feature Shapley value array ($feature\_shap$), keep ratio ($keep\_ratio$), parent 1 ($parent1$), parent 2 ($parent2$)
Output: Offspring 1 ($child1$), Offspring 2 ($child2$)
1: Compute the total number of features: $num\_features$
2: Sort the features in descending order of Shapley values, obtaining the indices: $sorted\_indices$
3: Calculate the number of top features to keep: $num\_keep = num\_features \times keep\_ratio$
4: Select the indices of the top Shapley value features: $keep\_features = sorted\_indices[: num\_keep]$
5: Select the indices of the remaining lower Shapley value features: $non\_keep\_features = sorted\_indices[num\_keep :]$
6: Initialize offspring: $child1$ and $child2$
7: Preserve high-importance features:
7.1 $child1[keep\_features] = parent1[keep\_features]$
7.2 $child2[keep\_features] = parent2[keep\_features]$
8: Perform crossover on low-importance features:
8.1 **for** each index i in $non\_keep\_features$ **do**
8.1.1 Randomly decide the values for $child1[i]$ and $child2[i]$ by selecting from $parent1[i]$ and $parent2[i]$
end for
9: **Return** offspring $child1$ and $child2$

Mutation operator

Traditional single-point mutation operations often overlook the importance of features and tend to randomly select a feature for mutation. This simple method lacks specificity and can result in key features being mutated, thereby affecting the individual's adaptability and overall performance. To overcome this drawback, this study proposes a mutation operation based on feature Shapley values. Specifically, the proposed mutation operation adjusts based on feature Shapley values. The initial mutation rate is adjusted according to the Shapley values, with features having lower Shapley values have a higher probability of mutation. Additionally, a sparsity factor is introduced to further influence the mutation probability. This mutation strategy enhances the diversity of candidate solutions while preserving important features. Algorithm 3 illustrates the proposed mutation operator.

Algorithm 3: Mutation Operator Based on Shapley Values

Input: Feature Shapley value array ($feature\_shap$), initial mutation rate ($initial\_mutation\_rate$), sparsity factor ($sparsity\_factor$), individual ($individual$)
Output: Mutated individual ($individual$)
1: Compute the total number of features: $n\_features$
2: Calculate the mutation probability for each feature:
  2.1 $mutation\_prob = initial\_mutation\_rate \times (1 - feature\_shap)$
  2.2 $mutation\_prob += sparsity\_factor \times (1 - feature\_shap)$
3: Randomly select a feature for mutation:
  3.1 Perform weighted random selection based on $mutation\_prob$, obtaining $index\_to\_mutate$
4: Mutation: Flip the selected feature:
  4.1 $individual[index\_to\_mutate] = 1 - individual[index\_to\_mutate]$
5: **Return** mutated individual $individual$

3.3 Quality inspection model based on SPE and tree modeling

To effectively address the challenges of class imbalance and class overlap in semiconductor manufacturing quality inspection, this study integrates the self-paced ensemble (SPE) framework with four tree-based machine learning algorithms: Decision Tree, Light Gradient Boosting Machine (LightGBM), Extreme Gradient Boosting (XGBoost), and Extra Trees. This integration aims to enhance the robustness and accuracy of quality classification under complex data distributions. The following subsections describe the structure of the SPE framework and the characteristics of the selected tree-based classifiers.

The framework of SPE

SPE is an adaptive ensemble framework designed for the classification of large-scale imbalanced data. It introduces the concept of classification hardness, which measures the difficulty of classifying a sample due to class imbalance, noisy samples, and class overlap. Different samples typically have varying levels of classification hardness. Noisy samples and boundary-overlapping samples usually exhibit higher classification hardness values, while common majority of samples tend to have lower hardness values. Different classifiers may find it more or less difficult to classify the same sample [14]. Therefore, the classification hardness value for the same sample can vary across different classifiers. For a given dataset, its classification hardness distribution is defined by the hardness values of all samples. This distribution can better guide resampling, thereby improving the model's classification performance.

The framework of SPE is illustrated in Fig. 1. The core of the SPE framework lies in combining classification hardness-coordinated undersampling techniques with a self-paced training process, forming a self-paced undersampling technique [14]. This method employs adaptive training techniques to manage classification difficulty while addressing sampling insufficiencies. In each iteration, the strategy selects the most informative negative samples based on classification difficulty. Through adaptive undersampling, the SPE algorithm incrementally focuses on classifying challenging instances while maintaining an understanding of the distribution of easily classified samples, thus reducing the risk of overfitting.

Decision Trees

Decision Trees are widely used in quality inspection tasks across various manufacturing domains, particularly for handling high-dimensional and multi-class feature data. In complex production environments, where numerous process uncertainties exist, Decision Trees can recursively partition the dataset and generate interpretable decision rules that help identify key factors influencing product quality. Their ability to model nonlinear relationships and provide transparent decision paths makes them especially suitable for quality analysis in complex manufacturing systems.

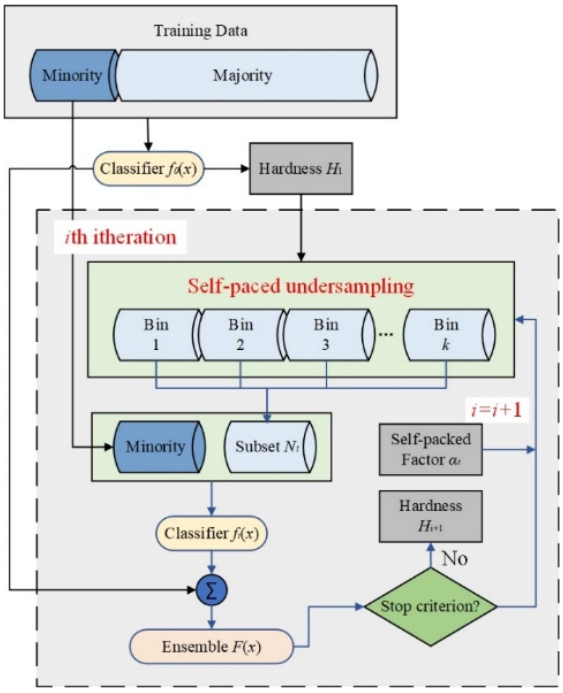

Fig. 1 The framework of SPE

XGBoost

XGBoost is an ensemble learning algorithm based on gradient boosting, capable of progressively building multiple weak learners to reduce error and employing regularization to prevent overfitting. Given the high dimensionality and class overlap issues in the semiconductor manufacturing dataset used in this study, XGBoost's strong generalization ability and capacity to handle complex feature spaces make it particularly suitable. Additionally, XGBoost's high computational efficiency and capability to handle missing values enable it to rapidly construct effective quality inspection models on large-scale data.

LightGBM

LightGBM excels in handling large-scale, high-dimensional data, particularly through its histogram-based decision tree learning approach, which significantly reduces computational resource consumption. Given the large number of high-dimensional features in semiconductor manufacturing, LightGBM's efficiency and scalability make it particularly suitable for this task. Compared to other methods, LightGBM employs a leaf-wise growth strategy, effectively increasing model training speed while ensuring robustness when facing imbalanced data.

Extra Trees

Extra Trees is an ensemble learning algorithm specifically designed to enhance the performance of decision trees. It achieves this by generating multiple unbiased trees and employing majority voting in the final prediction. In the context of quality inspection, Extra Trees can process large amounts of multidimensional data and capture complex feature relationships, significantly improving classification accuracy.

3.4 Evaluation metrics for quality inspection models

Quality inspection is a binary classification problem where defective instances are classified as the minority class and quality-compliant instances are classified as the majority class. In binary classification, the results are divided into four different groups: true positive (TP), false positive (FP), false negative (FN), and true negative (TN). To evaluate the performance of the quality inspection model under class imbalance conditions, this study employs key performance indicators such as Recall, Specificity, G-Mean, AUROC, and AUPRC.

Recall: Recall refers to the proportion of actual positive (quality-compliant) samples that are correctly predicted as positive by the model. It can be expressed using Eq. 6.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{6}$$

Specificity: Specificity refers to the proportion of actual negative (defective) samples that are correctly predicted as negative by the model. In quality inspection, specificity measures the model's accuracy in identifying defective samples. This is particularly important to ensure that defective instances are not missed. Specificity can be expressed using Eq. 7.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{7}$$

G-Mean: G-Mean is a metric used to evaluate the performance of classifiers on imbalanced datasets, especially in situations where the minority class holds greater importance. A higher G-Mean value indicates that the quality inspection model performs well in identifying both quality-compliant and defective products, thereby ensuring the model's overall performance on imbalanced datasets. G-Mean can be expressed using Eq. 8.

$$\text{G-mean} = \sqrt{\left(\frac{TP}{TP + FN}\right)\left(\frac{TN}{FP + TN}\right)} \tag{8}$$

AUROC – Area Under the Receiver Operating Characteristic Curve: AUROC measures the overall classification performance of the model. A higher AUROC value indicates better performance in distinguishing between positive and negative samples. In quality inspection, AUROC helps evaluate the model's performance across different decision thresholds.

AUPRC: AUPRC is particularly useful for imbalanced datasets. A higher AUPRC value indicates that the model can better balance precision and recall across different decision thresholds. In quality inspection, AUPRC reflects the model's ability to maintain high precision while reducing the false negative rate.

F1-score: F1-score is the harmonic mean of precision and recall. It provides a balanced measure of a model's performance by considering both false positives and false negatives. F1-score is especially important in quality inspection tasks where both types of errors—misclassifying defective items as compliant and vice versa—are costly. A higher F1-score indicates that the model maintains a good trade-off between precision and recall. It can be expressed using Eq. 9.

$$\text{F1-score} = \frac{2TP}{2TP+FP+FN} \tag{9}$$

3.5 Computational cost analysis

The proposed MOFS-SPE framework consists of two stages: offline MOEA/D-based feature selection and online SPE classifier training and prediction. In the MOEA/D feature selection stage, let G denote the number of generations, P the population size, n the number of samples, and k the number of selected features. Each candidate solution requires the evaluation of two objectives: AUPRC and the R-value. The computation of AUPRC has a complexity of $O(n \cdot k + n\,log\,n)$, while the R-value, based on kNN, requires $O(n^2 \cdot k)$. Therefore, the overall complexity of MOEA/D can be expressed as $O(G \cdot P \cdot [(n \cdot k + n \log n) + (n^2 \cdot k)])$. Since this procedure is performed offline, it does not affect real-time operations. In the SPE classifier stage, let I denote the number of self-paced iterations, T the number of trees in the final ensemble, and d the average tree depth. The training complexity is approximately $O(I \cdot (n^+ + n^-) \cdot k \cdot \log(n^+ + n^-))$ where n^+ and n^-

represent the minority and majority class sizes, respectively. The prediction complexity is $O(T \cdot d)$ which is sufficiently low to support real-time inference.

In summary, the MOFS-SPE framework follows an 'offline training + online prediction' paradigm. The offline stage is dominated by the kNN-based R-value computation, while the online prediction complexity remains $O(T \cdot d)$, thus ensuring that the computational cost meets the requirements of online quality prediction in semiconductor manufacturing.

4. Case study

To evaluate the effectiveness of the proposed quality inspection framework, experiments are conducted using a real-world semiconductor manufacturing dataset [28]. The dataset is first preprocessed to ensure data quality and consistency. Subsequently, a representative subset of features is selected using the proposed multi-objective feature selection algorithm based on MOEA/D. The quality inspection model is then constructed by integrating the SPE framework with four tree-based classifiers: Decision Tree, LightGBM, XGBoost, and Extra Trees. Finally, the performance of the proposed model is compared against several mainstream baseline methods to assess its effectiveness in addressing class imbalance and class overlap in quality inspection tasks. The programming work for the multi-objective feature selection and quality inspection model is completed in a Python 3 environment, utilizing the Scikit-learn library and the work referenced in [14].

4.1 Data preprocessing

The semiconductor component manufacturing process dataset contains 1567 samples, each with 590 features and 1 label. The features represent sensor parameters collected during the semiconductor manufacturing process, and the label indicates whether the product passes or fails the quality inspection (1 = pass, −1 = fail). As shown in Fig. 2, the dataset includes 1463 quality-compliant samples and 104 defective samples, resulting in a highly imbalanced dataset with an imbalance ratio of 14.06. Additionally, the dataset contains a large number of missing values, as illustrated in Fig. 3.

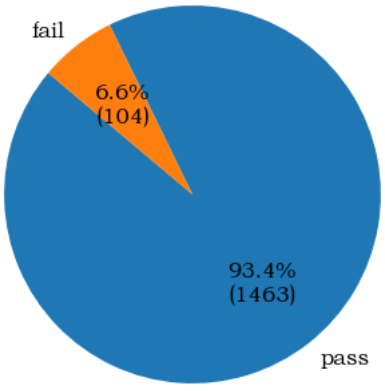

Fig. 2 Number of pass and fail samples

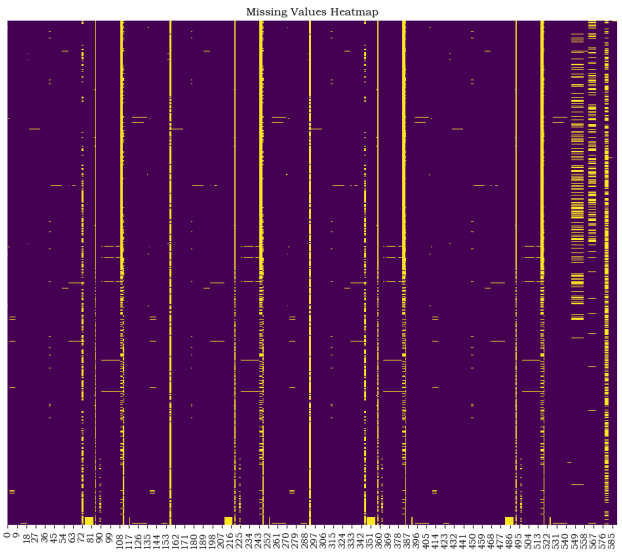

Fig. 3 Heatmap of missing values in the dataset

4.2 Feature selection

The objective of feature selection is to identify a compact subset of informative features that effectively represent quality-related patterns. In this subsection, the Shapley values for each feature are first calculated. Then, the proposed MOEA/D multi-objective feature selection algorithm is used to select a subset of features from the high-dimensional data that reduces class overlap and improves classification accuracy.

The proposed feature selection algorithm initializes, crosses, and mutates populations based on the Shapley values of features. Therefore, the Shapley values for each feature need to be calculated before feature selection. In this study, the LightGBM model is used to calculate Shapley values, assessing each feature's contribution to the model's prediction results. For calculating Shapley values, the LightGBM classification model's n_estimators is set to 1000 and max_depth to 10, with other parameters using Scikit-learn's default settings. The SHAP contributions of the top 10 important features are shown in Fig. 4. In the figure, the horizontal axis displays Shapley values, with positive Shapley values indicating a positive impact on the quality inspection model and negative Shapley values indicating a negative impact.

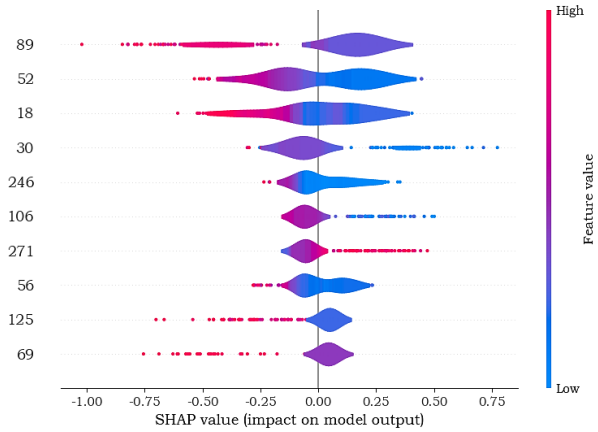

Fig. 4 SHAP contributions of the top 10 important features

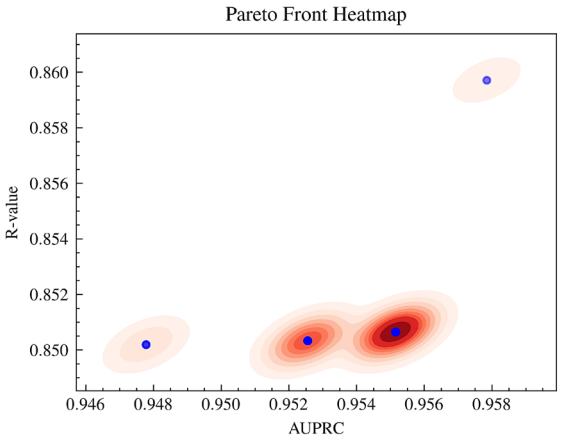

Fig. 5 Pareto solutions obtained by the MOEA/D feature selection algorithm

The population size for the MOEA/D feature selection algorithm is set to 100, with a maximum of 100 iterations and a crossover probability of 0.9. During population initialization and crossover, the retention ratio for important features is set to 0.3. The mutation probability is set to 0.1, and the sparsity factor for the mutation operator is set to 0.05. These settings follow common practices in MOEA/D studies and are consistent with recommendations in the literature [29]. The KNN-3 model is used as the evaluation model during feature selection. As shown in Fig. 5, the MOEA/D feature selection algorithm obtains four different Pareto solutions.

The LightGBM quality inspection model is trained and validated using the four selected feature sets obtained from the MOEA/D feature selection algorithm and the original, unselected features. The LightGBM classification model is configured with n_estimators set to 1000 and max_depth set to 10, with other parameters using the default settings from the Scikit-learn library. The evaluation metrics for the four feature selection schemes and the original feature scheme are shown in Table 1. As seen in Table 1, the second feature selection scheme chosen by the MOEA/D algorithm achieves the best performance across all evaluation metrics, achieving the highest values in Recall, AUROC, and AUPRC. Therefore, the second feature selection scheme is selected as the final scheme. Compared to the model using unselected features, the AUROC increases from 0.685 to 0.770, an improvement of 12.4 %, and the AUPRC increases from 0.932 to 0.972, an improvement of 4.3 %.

To evaluate the stability of the MOEA/D-based feature selection process, we repeated the optimization 10 times under identical settings. For each run, the feature subset with the highest AUPRC was selected and evaluated using the LightGBM classifier. Fig. 6 presents the AUPRC values of these best-performing subsets across runs. The results show that the AUPRC values consistently fall within the range of 0.970 to 0.983, indicating minimal performance variation.

Table 1 Evaluation metrics for different feature selection schemes

Features	Recall	AUROC	AUPRC
MOEA/D_1	1.0	0.747	0.965
MOEA/D_2	1.0	**0.770**	**0.972**
MOEA/D_3	1.0	0.735	0.967
MOEA/D_4	1.0	0.766	0.968
original	1.0	0.685	0.932

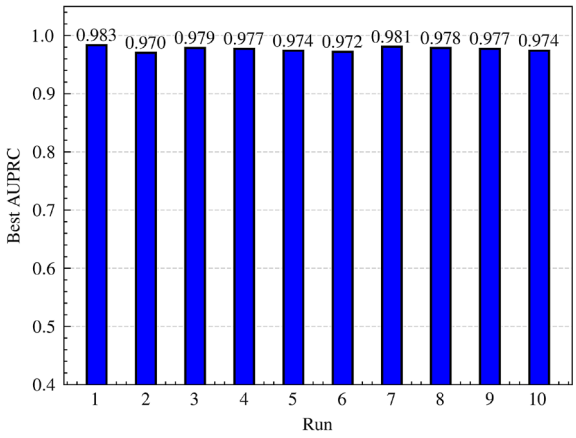

Fig. 6 AUPRC values of the best-performing feature subsets selected by MOEA/D across 10 independent runs

4.3 Quality inspection model construction

After obtaining the optimal feature subset, decision trees, XGBoost, LightGBM, and Extra Trees are used as base models within the SPE framework to construct the quality inspection models. To achieve the best performance for the quality inspection models, the Optuna [30] framework is employed to optimize the hyperparameters of the four base classifiers: decision tree, XGBoost, LightGBM, and Extra Trees. The hyperparameter optimization aims to maximize the AUROC, with 100 iterations. The ranges for hyperparameter optimization and the best-obtained hyperparameters for the base classifiers are shown in Table 2.

Then, the four hyperparameter-optimized base classifiers are integrated with the SPE framework, forming the SPE-DT, SPE-LightGBM, SPE-XGBoost, and SPE-ET models. The estimators parameter within the SPE framework is then optimized, using AUROC as the evaluation metric. The optimization range for the estimators parameter is [1, 300], and the AUROC values for the four models under different estimators are shown in Fig. 7. Finally, the optimal estimators for SPE-DT, SPE-LightGBM, SPE-XGBoost, and SPE-ET are 230, 100, 100, and 30, respectively.

Table 2 Hyperparameter ranges and best hyperparameters for different quality inspection models

Hyperparameters	Ranges	Decision tree	LightGBM	XGBoost	Extra trees
n_estimators	[200,2000]	N/A	1400	1900	650
max_depth	[5,40]	15	34	39	6
learning_rate	[0.01,0.5]	N/A	0.086	0.252	N/A
reg_lambda	[0,5]	N/A	0	N/A	N/A
reg_alpha	[0,5]	N/A	0	N/A	N/A

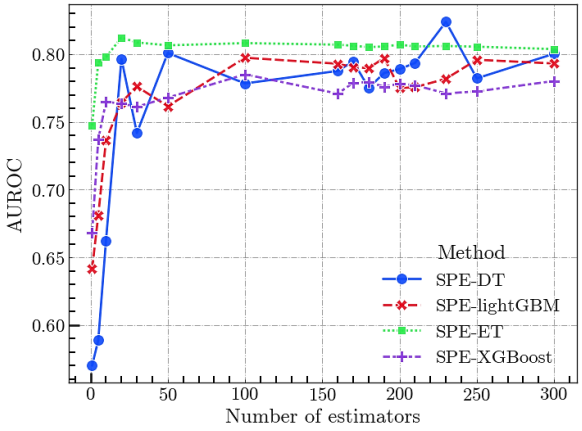

Fig. 7 AUROC for different estimators in four SPE models

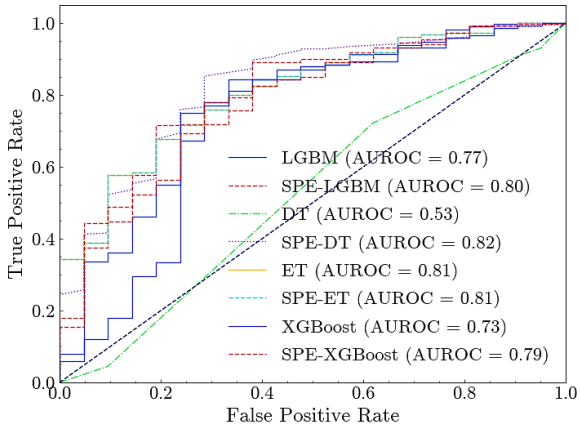

Fig. 8 ROC curves for different quality inspection models

The selected optimal feature subset is used to train the hyperparameter-optimized models, the ROC curves for each model are shown in Fig. 8, and the confusion matrices for each model are shown in Fig. 9. When using an independent decision tree model, only 1 defective product is

correctly identified, while 20 defective products are incorrectly identified as quality-compliant. As shown in Table 3, the independent DT model achieves a Specificity of 0.048, a G-Mean of 0.210, and an AUROC of 0.525. In contrast, when combining the SPE framework with the decision tree model, 14 out of 21 defective samples are correctly identified as defective by the SPE-DT model. Its Specificity reaches 0.667, G-Mean reaches 0.758, and AUROC reaches 0.824. The relative percentage difference in G-Mean between SPE-DT and DT is 261.90 %.

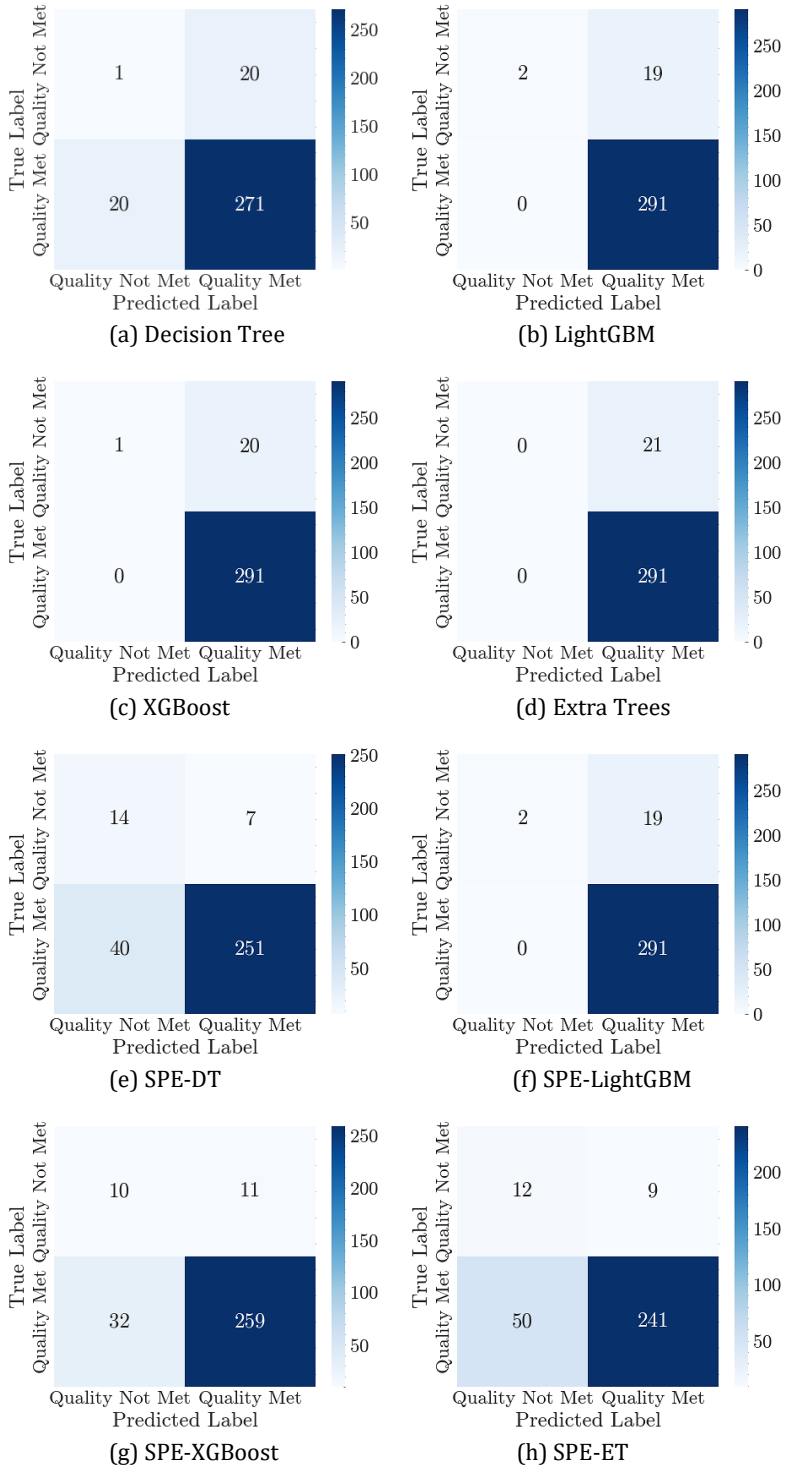

Fig. 9 Confusion matrices for different quality inspection models

Table 3 Performance evaluation of different three-based classifiers for spe quality inspection models

Classifiers	Recall	Specificity	G-Mean	AUROC	AUPRC
Decision tree	0.931	0.048	0.210	0.525	0.936
LightGBM	**1.0**	0.095	0.308	0.770	0.974
XGBoost	**1.0**	0.047	0.218	0.731	0.964
Extra Trees	**1.0**	0.0	0.0	0.784	0.979
SPE-DT	0.862	**0.667**	**0.758**	**0.824**	**0.983**
SPE-LightGBM	0.927	0.333	0.556	0.797	0.979
SPE-XGBoost	0.890	0.476	0.651	0.785	0.978
SPE-ET	0.828	0.571	0.688	0.809	**0.983**

The independent LightGBM model achieves a Specificity of 0.095, a G-Mean of 0.308, and an AUROC of 0.770. In contrast, when LightGBM is used as the base estimator within the SPE framework, the model's Specificity increases to 0.333, G-Mean to 0.556, and AUROC to 0.797. The relative percentage difference in G-Mean between SPE-LightGBM and the independent LightGBM model is 80.52 %.

The standalone XGBoost model correctly identifies only 1 defective product, while incorrectly identifying 20 defective products as quality-compliant. The independent XGBoost model achieves a Specificity of 0.047, G-Mean of 0.218, and AUROC of 0.731. In contrast, when XGBoost is used as the base estimator within the SPE framework the model correctly classifies 10 out of 21 defective samples, achieving a Specificity of 0.476, G-Mean of 0.651, and AUROC of 0.785. The relative percentage difference in G-Mean between SPE-XGBoost and the independent XGBoost model is 198.62 %.

Similarly, using the standalone Extra Trees model, all 21 defective samples are incorrectly classified as quality-compliant, resulting in a Specificity of 0, G-Mean of 0, and AUROC of 0.784. In contrast, when Extra Trees is used as the base estimator within the SPE framework, the model correctly classifies 12 out of 21 defective samples, improving the Specificity to 0.571, G-Mean to 0.688, and AUROC to 0.809.

Overall, the SPE framework significantly enhances the performance of all models. Among the evaluated models, the SPE-DT model performs the best. In addition, the performance improvements obtained across different tree-based classifiers indicate that the SPE framework exhibits a degree of robustness to the choice of base learner, although variations in absolute performance remain. However, the SPE model still exhibits some misclassification issues, mistakenly identifying some quality-compliant products as defective. In the context of quality inspection, it is important to recognize that the cost of incorrectly judging defective products as quality-compliant far outweighs the cost of misclassifying quality-compliant products as defective. The former can lead to defective products entering the market, resulting in customer complaints and potential recall risks, whereas the latter typically only triggers additional manual re-inspection and therefore has a lesser impact.

4.4 Comparison with other quality inspection models

To validate the effectiveness of the proposed quality inspection model, the best-performing SPE-DT model is compared with SmoteBoost, RusBoost, SmoteBagging, and BalanceCascade models. SmoteBoost combines Synthetic Minority Over-sampling Technique (SMOTE) and AdaBoost, making it particularly suitable for handling class imbalance issues. RusBoost integrates Random Under Sampling (RUS) and AdaBoost algorithms to balance the dataset by reducing majority class samples, enhancing the classifier's ability to identify minority class samples. SmoteBagging combines SMOTE and Bagging to improve the detection rate of minority class samples by performing SMOTE sampling on each subset followed by Bagging training, thereby improving the classifier's performance on imbalanced datasets. BalanceCascade gradually eliminates poorly performing features and samples through a layer-by-layer selection process, ultimately optimizing classification performance. These models are competitive approaches for addressing class imbalance and class overlap problems. In this study, the parameters for all models are consistent with their original settings in the respective papers.

In Table 4, the SPE-DT model is compared with other common quality inspection models. According to the results in Table 4, the SPE-DT model performs exceptionally well across the key

metrics of Specificity, G-Mean, and AUROC. The Specificity of the SPE-DT model reaches 0.667, significantly outperforming the other models. Furthermore, the improvement in G-Mean is particularly noteworthy. The SPE-DT's G-Mean value of 0.758 represents an approximate 27.2 % increase compared to RusBoost's 0.596 and a 34.9 % improvement over BalanceCascade's 0.562. Among all models, the SPE-DT achieves the highest G-Mean among all evaluated models, indicating a more balanced performance in terms of detection accuracy and recall. Regarding AUROC, the SPE-DT model achieves a value of 0.824, slightly lower than BalanceCascade's 0.827, yet still superior to most models. Overall, the significant improvements in Specificity and G-Mean, along with its competitive AUROC, demonstrate the superiority of the SPE-DT model in quality inspection.

Furthermore, to further validate the applicability and superiority of the proposed SPE-DT model in industrial quality-inspection tasks, we compared it against several representative methods from recent literature, including particle swarm optimization (PSO), naive bayes, RF, gradient deep learning boosting (GDLB), and contribution-degree-based spiking neural network (CDSNN). As shown in Table 5, SPE-DT outperforms all competing approaches on multiple key metrics: it achieves a recall of 0.862 and a specificity of 0.667, striking a strong balance between positive and negative class identification and yielding a G-Mean of 0.758, substantially higher than that of the other methods. Its F1-score of 0.858 further demonstrates an optimal trade-off between precision and recall, and an AUROC of 0.824 confirms its superior discriminative capability on imbalanced data.

Table 4 Performance evaluation of SPE-DT model compared to other models

Classifiers	Recall	Specificity	G-Mean	AUROC	AUPRC
SMOTEBoost	**1.0**	0	0	0.804	0.979
RusBoost	0.931	0.381	0.596	0.796	0.980
SMOTEBagging	**1.0**	0	0	0.812	0.980
BalanceCascade	0.948	0.333	0.562	**0.827**	0.981
SPE-DT	0.862	**0.667**	**0.758**	0.824	**0.983**

Table 5 Performance comparison between the proposed model and representative literature methods

Method	Recall	Specificity	G-Mean	F1-score	AUROC
PSO [31]	**0.986**	0.110	0.329	0.688	N/A
Naive Bayes [10]	N/A	**0.667**	N/A	N/A	0.747
RF [7]	0.380	N/A	N/A	0.290	0.720
RF [32]	N/A	N/A	N/A	N/A	0.770
GDLB [33]	0.690	N/A	N/A	0.310	0.790
CDSNN [34]	0.20	N/A	N/A	0.190	0.560
Proposed	0.862	**0.667**	**0.758**	**0.858**	**0.824**

5. Conclusion

This study presents a novel quality inspection framework, MOFS-SPE, that integrates multi-objective feature selection with a self-paced ensemble learning strategy to address the challenges of class imbalance and class overlap in semiconductor manufacturing. The framework employs a MOEA/D-based feature selection algorithm, using AUPRC and *R*-value as objective functions, to extract a compact and informative subset of quality-related features from high-dimensional process data. Subsequently, the self-paced ensemble framework is combined with four tree-based classifiers to construct a robust quality classification model. The proposed approach is validated on a real-world semiconductor manufacturing dataset, and the experimental results demonstrate the following:

- Compared to models without feature selection, the multi-objective feature selection method based on MOEA/D significantly improves model performance. Specifically, AUROC is enhanced from 0.685 to 0.770, an increase of 12.4 %, and AUPRC rises from 0.932 to 0.972, an increase of 4.3 %.
- The SPE framework significantly enhances the overall performance of the four tree models, particularly in reducing the misclassification of defective products. The SPE-DT model shows notable improvements in Specificity and G-Mean, and its competitiveness in AUROC highlights its superiority in quality inspection.

- Comparison with models such as SmoteBoost, RusBoost, SmoteBagging, and BalanceCascade indicates that the SPE-DT model excels in Specificity, G-Mean, and AUROC.
- The proposed method is essentially data-driven and does not explicitly incorporate process knowledge, which limits the interpretability of the selected features from a physical or engineering perspective. Future research will explore the integration of domain knowledge or physics-informed modeling to enhance the interpretability and practical relevance of the framework. In addition, assessing the applicability of the proposed method in actual production environments and real-time inspection settings will be an important direction for future work.

Acknowledgement

This research was supported by the Science and Technology Plan Project of Yulin City, China (Grant No. 2023-CXY-206), the Shaanxi Provincial Key R&D Program Project (Grant No. 2024CY2-GJHX-16), and the Doctoral Dissertation Innovation Fund of Xi'an University of Technology (Grant No. BC202603).

References

[1] Song, W.T., Huo, L. (2024). Machine learning for enhancing manufacturing quality control in ultrasonic nondestructive testing: A Wavelet Neural Network and Genetic Algorithm approach, *Advances in Production Engineering & Management*, Vol. 19, No. 3, 347-357, doi: 10.14743/apem2024.3.511.

[2] Nuhu, A.A., Zeeshan, Q., Safaei, B., Shahzad, M.A. (2023). Machine learning-based techniques for fault diagnosis in the semiconductor manufacturing process: A comparative study, *The Journal of Supercomputing*, Vol. 79, No. 2, 2031-2081, doi: 10.1007/s11227-022-04730-x.

[3] Kang, S., Kim, E., Shim, J., Chang, W., Cho, S. (2018). Product failure prediction with missing data, *International Journal of Production Research*, Vol. 56, No. 14, 4849-4859, doi: 10.1080/00207543.2017.1407883.

[4] He, Z., Hu, H., Zhang, M., Yin, X. (2022). Product quality prediction based on process data: Taking the gearbox of Company D as an example, *Quality Engineering*, Vol. 34, No. 3, 409-422, doi: 10.1080/08982112.2022.2063743.

[5] Carvajal Soto, J.A., Tavakolizadeh, F., Gyulai, D. (2019). An online machine learning framework for early detection of product failures in an Industry 4.0 context, *International Journal of Computer Integrated Manufacturing*, Vol. 32, No. 4-5, 452-465, doi: 10.1080/0951192X.2019.1571238.

[6] Jeong, E.-Y., Kim, J., Jang, W.-H., Lim, H.-C., Noh, H., Choi, J.-M. (2021). A more reliable defect detection and performance improvement method for panel inspection based on artificial intelligence, *Journal of Information Display*, Vol. 22, No. 3, 127-136, doi: 10.1080/15980316.2021.1876174.

[7] Kang, Z., Catal, C., Tekinerdogan, B. (2022). Product failure detection for production lines using a data-driven model, *Expert Systems with Applications*, Vol. 202, Article No. 117398, doi: 10.1016/j.eswa.2022.117398.

[8] Wang, N., Li, P., Zhao, Z., Yang, J. (2022). Quality abnormal recognition model based on convolutional neural network, *Operations Research and Management Science*, Vol. 31, 220-225, doi: 10.12005/orms.2022.0205.

[9] Hong, Y., Yang, M., Jiang, Y., Du, D., Chang, B. (2023). Real-time quality monitoring of ultrathin sheets edge welding based on microvision sensing and SOCIFS-SVM, *IEEE Transactions on Industrial Informatics*, Vol. 19, No. 4, 5506-5516, doi: 10.1109/TII.2022.3199258.

[10] Ismail, M., Mostafa, N.A., El-assal, A. (2022). Quality monitoring in multistage manufacturing systems by using machine learning techniques, *Journal of Intelligent Manufacturing,* Vol. 33, No. 8, 2471-2486, doi: 10.1007/s10845-021-01792-1.

[11] Santos, M.S., Abreu, P.H., Japkowicz, N., Fernández, A., Santos, J. (2023). A unifying view of class overlap and imbalance: Key concepts, multi-view panorama, and open avenues for research, *Information Fusion*, Vol. 89, 228-253, doi: 10.1016/j.inffus.2022.08.017.

[12] Hancock, J.T., Khoshgoftaar, T.M., Johnson, J.M. (2023). Evaluating classifier performance with highly imbalanced Big Data, *Journal of Big Data*, Vol. 10, No. 1, Article No. 42, doi: 10.1186/s40537-023-00724-5.

[13] Borsos, Z., Lemnaru, C., Potolea, R. (2018). Dealing with overlap and imbalance: A new metric and approach, *Pattern Analysis and Applications*, Vol. 21, No. 2, 381-395, doi: 10.1007/s10044-016-0583-6.

[14] Liu, Z., Cao, W., Gao, Z., Bian, J., Chen, H., Chang, Y., Liu, T.Y. (2020). Self-paced ensemble for highly imbalanced massive data classification, In: *Proceedings of 2020 IEEE 36th International Conference on Data Engineering (ICDE),* Dallas, USA, 841-852, doi: 10.1109/ICDE48307.2020.00078.

[15] Zhou, C., Fang, X. (2023). A convex two-dimensional variable selection method for the root-cause diagnostics of product defects, *Reliability Engineering & System Safety*, Vol. 229, Article No. 108827, doi: 10.1016/j.ress.2022.108827.

[16] Sun, C., Yin, Y., Kang, H., Ma, H. (2022). A distributed principal component regression method for quality-related fault detection and diagnosis, *Information Sciences*, Vol. 600, 301-322, doi: 10.1016/j.ins.2022.03.069.

[17] Liu, K., Hu, X., Zhou, H., Tong, L., Widanage, W.D., Marco, J. (2021). Feature analyses and modeling of lithium-ion battery manufacturing based on random forest classification, *IEEE/ASME Transactions on Mechatronics*, Vol. 26, No. 6, 2944-2955, doi: 10.1109/TMECH.2020.3049046.

[18] Saqlain, M., Jargalsaikhan, B., Lee, J.Y. (2019). A voting ensemble classifier for wafer map defect patterns identification in semiconductor manufacturing, *IEEE Transactions on Semiconductor Manufacturing*, Vol. 32, No. 2, 171-182, doi: 10.1109/TSM.2019.2904306.
[19] Mlinarič, J., Pregelj, B., Boškoski, P., Dolanc, G., Petrovčič, J. (2024). Optimization of reliability and speed of the end-of-line quality inspection of electric motors using machine learning, *Advances in Production Engineering & Management,* Vol. 19, No. 2, 182-196, doi: 10.14743/apem2024.2.500.
[20] Azamfar, M., Li, X., Lee, J. (2020). Deep learning-based domain adaptation method for fault diagnosis in semiconductor manufacturing, *IEEE Transactions on Semiconductor Manufacturing*, Vol. 33, No. 3, 445-453, doi: 10.1109/TSM.2020.2995548.
[21] Stavropoulos, P., Papacharalampopoulos, A., Stavridis, J., Sampatakakis, K. (2020). A three-stage quality diagnosis platform for laser-based manufacturing processes, *The International Journal of Advanced Manufacturing Technology*, Vol. 110, No. 11, 2991-3003, doi: 10.1007/s00170-020-05981-9.
[22] Chen, Z., Maske, H., Shui, H., Upadhyay, D., Hopka, M., Cohen, J., Lai, X., Huan, X., Ni, J. (2023). Stochastic deep Koopman model for quality propagation analysis in multistage manufacturing system, *Journal of Manufacturing Systems*, Vol. 71, 609-619, doi: 10.1016/j.jmsy.2023.10.012.
[23] Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W. (2003). SMOTEBoost: Improving prediction of the minority class in boosting, In: *Proceedings of PKDD 2003 – 7th European Conference on Principles and Practice of Knowledge Discovery in Databases,* Cavtat-Dubrovnik, Croatia, 107-119, doi: 10.1007/978-3-540-39804-2_12.
[24] Seiffert, C., Khoshgoftaar, T.M., Van Hulse, J., Napolitano, A. (2009). RUSBoost: A hybrid approach to alleviating class imbalance, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans,* Vol. 40, No. 1, 185-197, doi: 10.1109/TSMCA.2009.2029559.
[25] Wang, S., Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In: *Proceedings of 2009 IEEE Symposium on Computational Intelligence and Data Mining (CIDM),* Nashville, USA, 324-331, doi: 10.1109/CIDM.2009.4938667.
[26] Liu, X.-Y., Wu, J., Zhou, Z.-H. (2009). Exploratory undersampling for class-imbalance learning, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 39, No. 2, 539-550, doi: 10.1109/TSMCB.2008.2007853.
[27] 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,* Long Beach, USA, 4766-4777.
[28] McCann, M., Johnston, A. (2008). SECOM dataset UCI machine learning repository, from *https://archive.ics.uci.edu/dataset/179/secom*, accessed October 20, 2024.
[29] Pruvost, G., Derbel, B., Liefooghe, A., Li, K., Zhang, Q. (2020). On the combined impact of population size and sub-problem selection in MOEA/D, In: Paquete, L., Zarges, C. (eds.), *Evolutionary computation in combinatorial optimization*, *EvoCOP 2020, Lecture notes in computer science*, Vol 12102, Springer, Cham, Switzerland, 131-147, doi: 10.1007/978-3-030-43680-3_9.
[30] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD),* Anchorage, USA, 2623-2631, doi: 10.1145/3292500.3330701.
[31] Moldovan, D., Anghel, I., Cioara, T., Salomie, I. (2020). Particle swarm optimization based deep learning ensemble for manufacturing processes. In: *Proceedings of 2020 IEEE 16th International Conference on Intelligent Computer Communication and Processing (ICCP),* Cluj-Napoca, Romania, 563-570, doi: 10.1109/iccp51029.2020.9266269.
[32] Chazhoor, A., Mounika, Y., Vergin Raja Sarobin, M., Sanjana, M.V., Yasashvini, R. (2020). Predictive maintenance using machine learning based classification models, *IOP Conference Series: Materials Science and Engineering*, Vol. 954, No. 1, Article No. 012001, doi: 10.1088/1757-899X/954/1/012001.
[33] Nguyen, D.-K., Chan, C.-L., Phan, D.-V. (2022). Gradient deep learning boosting and its application on the imbalanced datasets containing noises in manufacturing, In: *Proceedings of 2021 International Conference on Security and Information Technologies with AI, Internet Computing and Big-data Applications,* Cham, Switzerland, 225-235, doi: 10.1007/978-3-031-05491-4_23.
[34] Liu, F., Yang, J., Pedrycz, W., Wu, W. (2022). A new fuzzy spiking neural network based on neuronal contribution degree, *IEEE Transactions on Fuzzy Systems*, Vol. 30, No. 7, 2665-2677, doi: 10.1109/tfuzz.2021.3090912.