INTEGRATION, the VLSI journal 104 (2025) 102482

Contents lists available at ScienceDirect

Integration

journal homepage: www.elsevier.com/locate/vlsi

Soft fault diagnosis in analog electronic circuits using supervised machine learning

M.I. Dieste-Velasco

Electromechanical Engineering Department, Higher Polytechnic School, University of Burgos, 09006, Burgos, Spain

ARTICLE INFO

Keywords:
Soft fault diagnosis
Fault classification
Machine-learning
Electronic circuits

ABSTRACT

Analog circuits are commonly used in a wide range of industrial applications, and their assessment is of great importance to ensure proper functionality and prevent faults. However, this task is not as fully developed and is significantly less advanced compared to the assessment of digital circuits, as soft faults are particularly difficult to detect in analog circuits. This study addresses the application of supervised classification techniques for the detection and classification of soft faults in analog circuits. A feature extraction methodology is proposed based on voltage measurements at key circuit points and across different frequencies, enabling precise characterization of system behavior. From this feature, a benchmark employing different machine learning methods was used. The evaluated classifiers include k-Nearest Neighbors (KNN), Naïve Bayes (NB), Discriminant Analysis Classifier (DAC), Classification Decision Tree (CDT), Random Forest (RF), Support Vector Machines (SVM) and Artificial Neural Networks (ANN). Each model was optimized through hyperparameter tuning and validated using cross-validation techniques. The results indicate that ANN and SVM achieved the best performance, attaining an accuracy of 97.92 % and 97.22 % on test data, with a global Matthews Correlation Coefficient (MCC) of 97.76 % and 97.01 %, respectively. Although RF obtained the highest training accuracy (99.39 %), its performance significantly dropped during testing (93.06 %, MCC of 92.52 %), indicating overfitting. Additionally, models such as KNN and DAC demonstrated solid performance, whereas NB and CDT were the least effective. These findings highlight the importance of carefully selecting both the feature set and the classification model for fault detection in electronic circuits. A Sallen-Key band-pass filter was used as the circuit under test (CUT), as soft fault classification in this type of circuit is particularly challenging. This study demonstrates that it is possible to accurately predict faults in circuits similar to the one analyzed.

1. Introduction

In the present work, supervised classification techniques will be employed, such as k-Nearest Neighbors (KNN), Naïve Bayes (NB), Discriminant Analysis Classifier (DAC), Classification Decision Tree (CDT), Random Forest (RF), Support Vector Machines (SVM), and Artificial Neural Networks (ANN), to model faults and predict the different soft fault scenarios that may occur in analog circuits.

As is well known, KNN classifies potential faults based on proximity to its nearest neighbors in the feature space, making it a simple but powerful technique for problems with complex decision boundaries. Likewise, Naïve Bayes (NB) classifiers are based on probability, assuming feature independence. Another method used in this work is the Discriminant Analysis Classifier (DAC), which uses linear or quadratic discriminant functions to separate classes. Additionally, this study analyzes the Classification Decision Tree (CDT), which constructs hierarchical trees based on sequential feature splits and the Random Forest (RF), which combines multiple decision trees. On the other hand, SVM are algorithms that find an optimal hyperplane to separate classes in the feature space, being effective in both linear and nonlinear problems through the use of kernel functions. Finally, ANN will be employed, consisting of one or multiple layers of interconnected neurons capable of learning complex and nonlinear relationships in the data. Furthermore, this study will perform a benchmark analysis among the different techniques analyzed and will present the results obtained for the determination and classification of soft faults in electronic circuits.

The main difference between classifiers such as KNN, NB, DAC, CDT, RF, and SVM, compared to artificial neural networks (ANN), lies in their ability to model complex relationships and their learning approach. Methods such as KNN, DAC, NB, and SVM rely on the geometry of the feature space or on statistical assumptions (such as independence in NB or a Gaussian distribution in DAC). Decision trees (CDT) and Random

E-mail address: midieste@ubu.es.

https://doi.org/10.1016/j.vlsi.2025.102482
Received 11 April 2025; Received in revised form 25 June 2025; Accepted 10 July 2025
Available online 11 July 2025

Forest (RF) are based on hierarchical rules, making them effective for structured problems, while SVM uses optimal margins to separate classes in both linear and nonlinear problems. In contrast, ANN, with one or multiple layers and nonlinear activation functions, can learn extremely complex and nonlinear patterns within the data, but at the cost of greater computational complexity. The aforementioned techniques will be evaluated comparatively in this study to analyze their performance in detecting and classifying soft faults in analog circuits.

Likewise, the study presents a methodology for feature extraction in the circuit for the detection of soft faults in analog circuits, consisting of extracting features from the circuit by measuring voltage at a series of predetermined points in the circuit and at different frequencies. This can be done with relative ease and is easily automated, making it possible to determine the circuit's behavior both in its nominal operating mode and in a fault situation. Likewise, the performances of the different analysis methods used for fault characterization in analog circuits will be evaluated, which is of great technological interest since analog circuits are used in a large number of high-responsibility engineering applications, and ensuring their proper operation is of utmost importance.

2. Review of the state of the art

Over the last few years, machine learning techniques such as KNN (k-Nearest Neighbors), NB (Naive Bayes), RF (Random Forest), Discriminant Analysis Classifier (DAC), Classification Decision Tree (CDT), Support Vector Machines (SVM) and Artificial Neural Networks (ANN), among others, have gained increasing popularity for classifying and detecting faults in analog electronic circuits. These techniques are particularly well-suited where reliability and fault detection are required. The growing use of machine learning in these fields is due to its ability to handle complex data patterns, improve diagnostic accuracy, and reduce the need for manual intervention. Analog circuits are widely used in many industrial systems and avionics [1]. Therefore, fault determination is very important to ensure their correct functioning. In the review by Afacan et al. [2], recent advancements in machine learning (ML) techniques for analog and radio frequency integrated circuit (IC) design are discussed, highlighting how ML-based methods are being applied across various design stages, from modeling and synthesis to layout and fault diagnosis. Some of the faults that may occur in analog electronic circuits include hard faults, soft faults, and intermittent faults, among others. In the research study of Qu et al. [1], the authors employed variational modal decomposition (VMD) and an autoencoder to detect intermittent faults in analog circuits. They then used adaptive dynamic density peak clustering to automatically classify fault types. Among their findings, they showed that VMD outperforms wavelet packet transform (WPT) and empirical mode decomposition (EMD) in detecting intermittent faults under noisy conditions. Likewise, in Fang et al. [3] a prior knowledge-guided teacher-student model was employed to detect intermittent faults (IFs) in analog circuits. Additionally, in Wang et al. [4] can be found an incipient fault diagnosis method for analog circuits which integrates multi-scale feature extraction and multi-channel feature fusion to enhance fault information completeness. Deep extreme learning machine denoising auto-encoder was used in their study for unsupervised feature extraction and fusion.

As is well known, the k-Nearest Neighbors (KNN) algorithm classifies data based on the majority of its k closest neighbors in the feature space through supervised training. It has the advantage of not requiring explicit training since it stores training data and evaluates distances such as Euclidean, Mahalanobis, or Manhattan, among others. Its performance depends on the k selection, the distance metric, and data distribution. It is highly effective in pattern recognition problems but can be computationally costly for large datasets. Among the research studies dealing with KNN, it is worth mentioning that of Tang and Xu [5], who employed KNN and conventional kernel density estimation (KDE) for fault classification in analog circuits. In their study, the cumulative influences on a datum from its neighbors corresponding to different classes were estimated using a Gaussian kernel function; and that of Sun et al. [6] where the authors employed wavelet packet energy spectrum and sparse random projections as preprocessing techniques to extract features from the circuit, and then they applied KNN for fault classification.

A machine learning approach for fault detection and classification in low-voltage DC microgrids, which combined a bagged ensemble learner and cosine k-Nearest neighbor (C-KNN) algorithms, was used by Deb and Jain [7] to identify and classify faults in a standalone low-voltage DC microgrid. Similarly, in Zare et. [8], a method which combined radial basis function (RBF) neural networks with machine learning was used to detect and classify faults in photovoltaic (PV) arrays. Likewise, in Madeti and Singh [9] a fault detection and classification technique for PV systems was proposed, utilizing a k-Nearest Neighbors (KNN) algorithm to detect and classify different types of faults. A KNN approach was also used for fault classification and localization in distribution networks with multiple distributed generators (DGs) in the study by Awasthi et al. [10], demonstrating high accuracy in fault identification.

Likewise, in recent years, there have been a large number of studies on fault determination in ICs based on machine learning techniques, as shown in the review by Roy et al. [11]. These approaches have been applied to testing analog, radio frequency, digital, and memory circuits, focusing on addressing the complexity of fault diagnosis using methods such as Artificial Neural Networks (ANN) and Principal Component Analysis (PCA). Some other methods for fault diagnosis in analog circuits can be found in Zhang and Li [12] who defined an output response consisting of a square matrix whose elements may vary depending on the circuit fault and, hence, they diagnosed the faults by comparing the faulty state with the normal behavior of the circuit. On the other hand, Shi et al. [13] proposed a method that employed density peaks clustering and a dynamic weight probabilistic neural network for analog circuit fault diagnosis, using an operational amplifier active filter as the circuit under test and in Shi et al. [14] a fault diagnosis method for analog circuits was shown by using Density Peak Clustering (DPC) and a Voting Probabilistic Neural Network (VPNN), combined with KNN.

Several machine learning algorithms were employed by Sudha et al. [15] to detect short-circuit faults in distribution transformers. Various feature extraction and classification techniques were evaluated, with k-Nearest Neighbor (KNN) identified as more effective than other methods, in terms of accuracy and processing time, including Quadratic Discriminant Analysis (QDA), Naïve Bayes (NB), and Linear Discriminant Analysis (LDA). A Naïve Bayes classifier combined with image-oriented feature extraction and selection techniques was employed by He et al. [16] for fault diagnosis in analog circuits. The method applied cross-wavelet transform to obtain time-frequency representations of fault signals, followed by feature selection using linear discriminant analysis. In another study, Arabi et al. [17] presented a machine learning-based method for identifying and categorizing parametric faults in analog circuits, utilizing frequency response characteristics of output voltage and supply current for feature extraction. After evaluating multiple classifiers, the quadratic discriminant classifier was selected for its superior accuracy. The feature set was generated using OrCAD PSpice and Monte Carlo analysis to simulate the circuits under test. In Silva et al. [18], autoencoders were used for data preprocessing and eleven algorithms were analyzed for fault classification in power distribution systems. Results showed that k-Nearest neighbor (KNN) and random forest (RF) achieved the best performance. Another method for detecting faults in analog circuits using cross-entropy between the fault-free and faulty circuit states was proposed by Li and Xie [19], based on the autoregressive (AR) model and Monte Carlo simulations to vary component values within tolerances. Likewise, Li et al. [20] proposed a method for diagnosing soft faults in nonlinear analog circuits through a feature fusion technique that integrated canonical correlation analysis and support vector machine (SVM).

Regarding ANN and ANFIS, Noussaiba and Abdelaziz [21] proposed an ANN-based fault diagnosis approach for induction motors using a

Multi-Layer Perceptron Neural Network to estimate stator inter-turn short-circuit severity. A fault diagnosis approach for analog electronic circuits using a Sugeno fuzzy logic classifier, based on statistical analysis of the circuit's frequency response to detect and identify faulty components, was proposed by Nasser et al. [22]. Similarly, Arabi et al. [23] proposed a fault classification approach for analog integrated circuits using a multiclass Adaptive Neuro-Fuzzy Inference System (ANFIS). Tadeusiewicz and Hałgas [24] proposed a method for diagnosing multiple soft faults in analog linear circuits, solving a least squares optimization problem using the Levenberg-Marquardt algorithm and a fault detection and isolation method for analog circuits using convolutional neural networks (CNNs), spectrogram-based feature extraction, and Monte Carlo simulations to generate signal samples for different faults was employed by Moezi and Kargar [25]. Further studies can be found in Binu et al. [26], which reviewed publications from the past few years on fault diagnosis in analog circuits. Their work focused on presenting a taxonomy of detection techniques, analyzing state-of-the-art methods, identifying research challenges, and highlighting the growing trend toward the adoption of machine learning techniques. Additionally, in Khemani et al. [27], a design of experiments-based approach was introduced to reduce fault classification complexity in analog circuits, integrating a wavelet-based deep learning network for fault analysis. Some other studies such as of Sheikhan and Sha'bani [28] analyzed the employment of ANN and Particle Swarm Optimization (PSO) in fault detection in analog circuits. Likewise, Zhao et al. [29] proposed an analog circuit fault diagnosis method that integrated Ensemble Empirical Mode Decomposition for feature extraction, the Maximum Information Coefficient for feature selection, and Particle Swarm Optimization to optimize Support Vector Machine (SVM) classification. Likewise Dieste-Velasco [30] employed a pattern recognition ANN for hard faults detection and Zhong et al. [31] used deep belief neural networks to detect intermittent faults in analog circuits. A dual-input model based on a multi-scale self-normalizing convolutional neural network was proposed by Yang et al. [32] to detect faults using circuit response signals and a fully convolutional network (FCN) was employed by Miao et al. [33] as a fault diagnosis model for analog circuits, using a global average pooling layer to determine fault category probabilities.

Further examples of ML to detect faults in several engineering applications can be found in the study by Shi et al. [34], who applied the Latent Dirichlet Allocation (LDA) topic model to extract features from railway signal equipment fault records and then used a Support Vector Machine (SVM) classifier for fault diagnosis, comparing its performance with Naïve Bayes (NB), Logistic Regression (LR), Random Forest (RF), and k-Nearest Neighbors (KNN). Another relevant study is the study by Fazli and Poshtan [35], who proposed a fault detection and isolation (FDI) method for wind turbines (WTs) using the k-Nearest Neighbors (KNN) classifier based on SCADA data, and the study by Chahal et al. [36], who studied stability prediction for smart energy grids using machine learning (ML) models, including Naïve Bayes, Decision Tree, Support Vector Machine, Random Forest, k-Nearest Neighbors, and Artificial Neural Networks (ANNs), among others. They found that ANNs optimized with the Adam optimizer achieved the highest accuracy, outperforming all other predictive models. Additionally, Afia et al. [37] investigated k-Nearest Neighbors (KNN), Ensemble Tree (ET), Multi-Class Support Vector Machine (MSVM), and Random Forest (RF), among others, for fault diagnosis based on motor current signal analysis and vibration analysis.

On the other hand, a fuzzy classifier-based technique for diagnosing single and multiple soft faults in analog electronic circuits was proposed by Kumar and Singh [38], where parameters like peak gain, frequency, and phase were extracted to differentiate between normal and faulty states, and a fault detection approach for analog circuits using an Extreme Learning Machine (ELM) optimized with the Firefly-Chaos Algorithm was employed by Yu et al. [39]. Likewise, Parai et al. [40] analyzed fault diagnosis in analog circuits by integrating output responses from multiple input signals and applying data fusion with Principal Component Analysis (PCA) and SVM classification. Bilski [41] proposed a hierarchical two-stage classification approach using self-organizing maps to separate easy and difficult fault cases, followed by Random Forest (RF) for complex cases. Zhao et al. [42] introduced a fault diagnosis method based on Deep Belief Networks (DBN), and Zhang et al. [43] proposed a wavelet transform-based feature extraction method combined with a Multiple Kernel Extreme Learning Machine (MKELM) for diagnosing analog circuit faults, where the extracted features were used to train an MKELM model with its parameters optimized via Particle Swarm Optimization (PSO), among many others.

This study is organized as follows: Section 3 describes the methodology, including the machine learning algorithms and feature extraction techniques applied to the circuit under test (CUT). Section 4 presents the classification results and compares the performance of the models. Section 5 discusses the key findings of this study, while Section 6 provides the conclusions and suggests directions for future research.

3. Methodology

The selection of predictor (independent) variables and the classification methods used in this study for fault detection in analog circuits are briefly described in this section. Matlab™ 2022b, as well as the Statistics and Machine Learning Toolbox of Matlab™ 2022b [44] and the Deep Learning Toolbox of Matlab™ 2022b [45], will be used in this study. More specifically, the Statistics and Machine Learning Toolbox is used to implement various supervised classification algorithms, such as Support Vector Machines (SVM), k-Nearest Neighbors (KNN), Naive Bayes (NB), Discriminant Analysis Classifier (DAC), Classification Decision Tree (CDT), and Random Forest (RF), and the Deep Learning Toolbox is employed for developing and training artificial neural networks (ANNs). Matlab™ 2022b itself serves as the platform to integrate these tools.

3.1. Selection of measurement points in the circuit

To determine the soft faults that appear in the circuit, a second-order band-pass filter will be selected as the circuit under test (CUT), in which fault determination is challenging due to the existence of common characteristics in the input variables used for fault classification. In order to obtain data on variables that allow the classification of soft faults, that is, deviations in circuit components that cause performance degradation without resulting in a complete failure, a Monte Carlo analysis will be used, taking into account the tolerances of the circuit components as well as situations in which these components deviate from their nominal value. As previously mentioned, a Sallen-Key band-pass filter will be used as the CUT, whose electrical schematic is shown in Fig. 1. By measuring at only two points in the circuit, at different frequencies, it is possible to obtain a dataset that will be used to determine the possible soft faults that may appear in the circuit. In the circuit shown in Fig. 1, commercial components with 5 % tolerances have been used, which will result in significant variations in the circuit, making fault determination difficult, as output responses may exhibit common characteristics due to circuit tolerance variations. This occurs because the circuit is sensitive to component tolerances, and therefore, both the quality factor and the frequency response of the circuit will vary. For example, the center frequency will change as it directly depends on the filter parameters. The dataset used for soft fault detection in this type of circuit has been selected from the two measurement points (OUT and M1) shown in Fig. 1, obtaining voltage measurements at the center frequency of the filter and at frequencies located at ±3 dB from the nominal frequency of the filter. Therefore, the proposed method consists of six predictors, represented by voltage measurements. These measurements act as inputs to the model and provide information that will be used to characterize the system state (nominal operation or some type of fault). The output corresponds to 15 different classes that describe the characteristics of possible faults in the circuit, that is, the nominal value

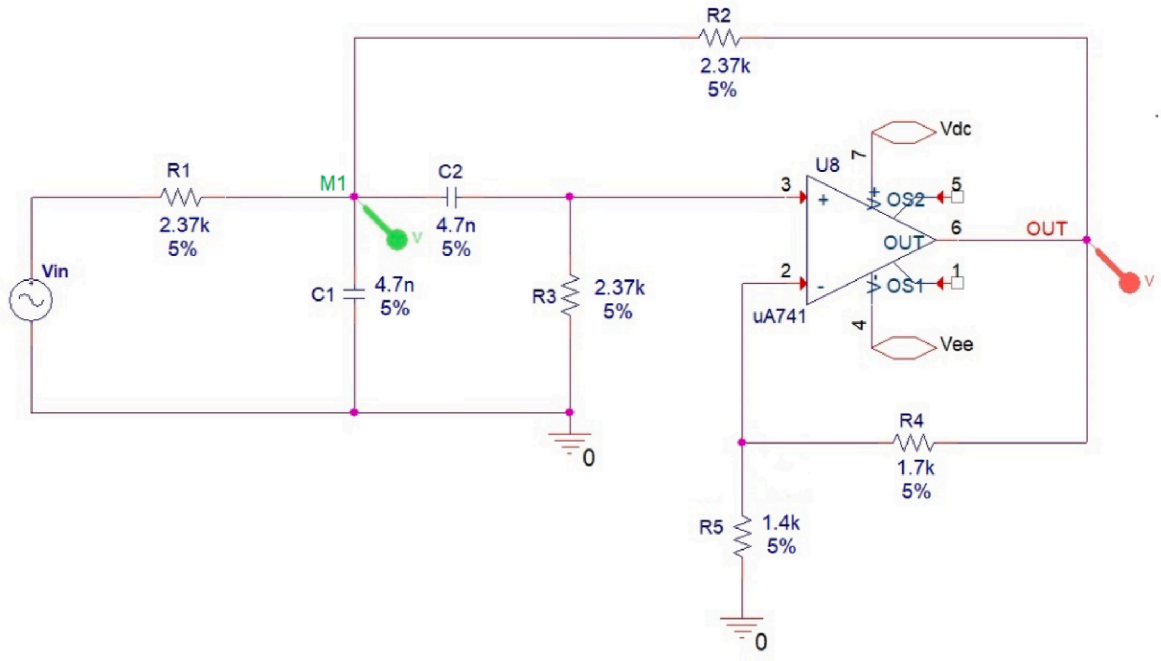

Fig. 1. Electrical diagram of the Sallen-Key band-pass filter employed as the CUT for determining soft faults.

and fourteen soft faults associated with variations in specific components. Based on this, it will be possible to train the supervised learning methods employed in this study: KNN, NB, DAC, CDT, RF, SVM, and ANN, so they can be applied to accurately detect anomalies in circuits. The capability of each of these models to detect the possible faults considered in this study will be demonstrated.

Table 1 shows the variation ranges of the circuit components in relation to their nominal values, that is, the variation ranges related to soft faults. To obtain the Monte Carlo values, it is assumed that the components vary, each within the ranges shown in Table 1, following a uniform distribution law, where the class names will be {'Nominal', '$C1_{low}$', '$C1_{high}$', '$C2_{low}$', '$C2_{high}$', '$R1_{low}$', '$R1_{high}$', '$R2_{low}$', '$R2_{high}$', '$R3_{low}$', '$R3_{high}$', '$R4_{low}$', '$R4_{high}$', '$R5_{low}$', '$R5_{high}$'}. As can be observed, deviations below the allowed values for the nominal values (low) and above them (high) are considered.

Based on the values shown in Table 1, a Monte Carlo analysis using Cadence® OrCAD® is carried out in which the nominal value of the circuit shown in Fig. 1 is replaced with the value corresponding to the soft fault to be analyzed. Once the features of the circuit have been extracted, MATLAB™ 2022b is used to develop supervised classification models to predict future fault situations as well as nominal values. Fig. 2 jointly represents the circuit responses corresponding to the frequency response of the Sallen-Key circuit at the selected points OUT and M1, using nominal values and those corresponding to each of the soft faults, when these situations are analyzed at the central value of the fault, i.e., without carrying out the Monte Carlo analysis. As can be observed in Figure 2, fifteen curves are represented for both OUT and M1, each corresponding to the nominal value of the circuit and to each of the soft faults. Likewise, the points where features will be extracted from the CUT are graphically indicated. In this study, these have been reduced to two measurement points at three different frequencies each, which are represented with dashed lines in the figure. As previously mentioned, these frequency values correspond to the nominal frequency and the frequencies located at ±3 dB from the nominal frequency. Only the value corresponding to the central value of these components is shown, not the Monte Carlo results, which are depicted in Fig. 3. From this figure, it can be observed the complexity of determining which type of fault corresponds to each output, based on the extracted features, as the values are very close to each other. For this reason, the determination of soft faults in analog circuits is a highly complex issue that has not yet been fully resolved.

Table 1
Variation ranges of the circuit components for soft fault analysis.

C1↓	[3.06 – 3.35] (nF)	R2↑	[2.96 – 3.20] (kΩ)
C1↑	[5.88 – 6.35] (nF)	R3↓	[1.54 – 1.69] (kΩ)
C2↓	[3.06 – 3.35] (nF)	R3↑	[2.96 – 3.20] (kΩ)
C2↑	[5.88 – 6.35] (nF)	R4↓	[1.11 – 1.21] (kΩ)
R1↓	[1.54 – 1.69] (kΩ)	R4↑	[2.13 – 2.30] (kΩ)
R1↑	[2.96 – 3.20] (kΩ)	R5↓	[0.91 – 1.00] (kΩ)
R2↓	[1.54 – 1.69] (kΩ)	R5↑	[1.75 – 1.89] (kΩ)

Fig. 3 shows the results obtained from the Monte Carlo analysis for the nominal case of the circuit, as well as for soft faults observed at the output (V_{OUT}) and at point M1 (V_{M1}), specifically for the nominal condition and the $C1_{low}$ fault scenario. According to the procedure described above, the voltage values at each of the three selected frequencies are extracted from these graphs. A similar procedure was followed for the remaining soft fault cases.

Fig. 4 shows the distribution of the different classes as a function of the three selected variables: V_{OUT}, V_{M1}, y $V_{OUT\text{-}3dB}$, which were chosen because they exhibited the highest correlation (V_{OUT} and V_{M1}) with the selected classes (soft faults) and the highest variance (V_{OUT} y $V_{OUT\text{-}3dB}$) in the dataset. As can be observed, regarding C1high and C2high, these classes have high values in V_{OUT}, which clearly separates them from the other categories. Regarding C2low and C1low, they are well clustered with lower values of V_{OUT} and $V_{OUT\text{-}3dB}$, making them easily distinguishable. On the other hand, with three variables, some classes remain partially separated, such as the Nominal class, which, although located in an intermediate region, is not perfectly separated, as it is relatively close to classes such as R1high and R2low. This could introduce some confusion in the classification models. R1low and R1high are close to each other but show some separation due to V_{OUT}. However, the overlap in V_{M1} could hinder precise classification.

Likewise, some classes with greater overlap can be observed such as R4low and R5low, which exhibit significant dispersion in $V_{OUT\text{-}3dB}$ y V_{M1}. This suggests that the selected variables are not sufficient to completely differentiate them. R3low and R2high also show significant proximity, especially in V_{OUT}, which could complicate their separation.

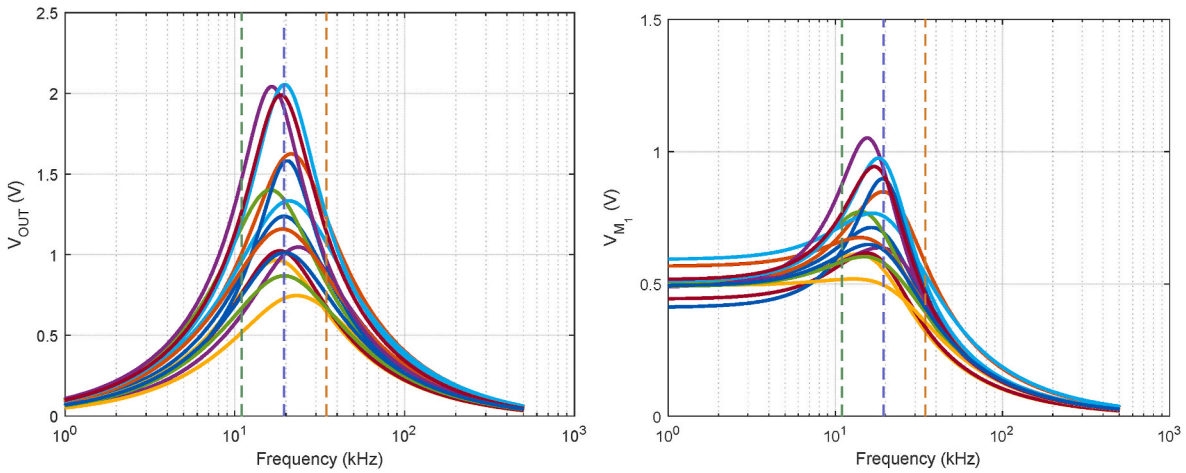

Fig. 2. Frequency response of the Sallen-Key circuit at OUT and M1 for soft faults and nominal behavior.

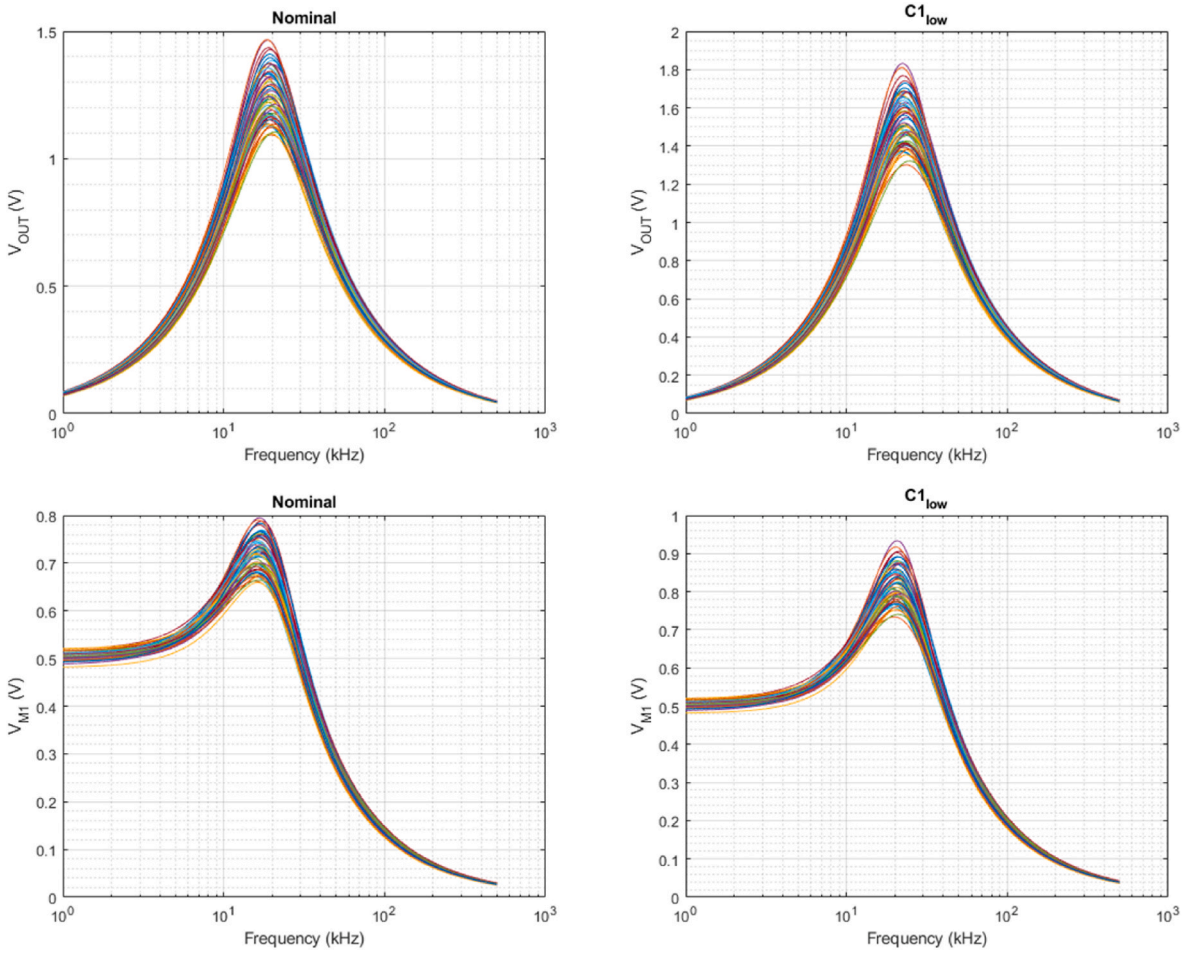

Fig. 3. Monte Carlo analysis results for the nominal case and the $C1_{low}$ fault, showing voltage responses at V_{OUT} and V_{M1}.

In light of the above, it is evident that fault prediction in this type of circuit is challenging. Based on the above, the effectiveness of the classification models used in the benchmarking conducted in this study will be analyzed, employing six variables, which, as previously mentioned, correspond to measurements at two points in the circuit (V_{OUT} and V_{M1}) and at three different frequencies (central and ± 3 dB). This will demonstrate that some of the analyzed models can achieve a high degree of discrimination between the different soft faults and the nominal behavior of the circuit.

3.2. Dataset structure and distribution

In the case of ANNs, the dataset will be divided into three main sets: 70 % for training, to adjust the network's weights and biases by minimizing the loss function during training, 15 % for validation, to monitor the model's performance during training and prevent overfitting, and

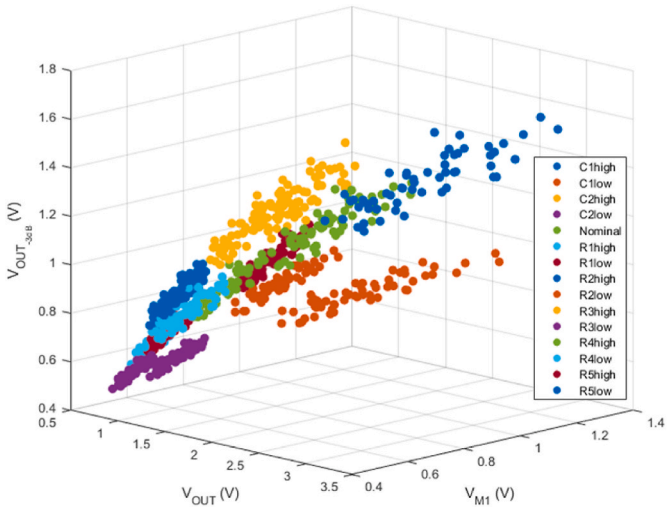

Fig. 4. Separation of nominal and soft faults values vs. the three most influential variables.

15 % for testing, to evaluate the model's performance on independent test data. Since other supervised classification techniques will also be used, cross-validation will be applied to these classification techniques so that the comparison of the obtained results is conducted on a similar dataset. Specifically, cross-validation divides the dataset into k subsets (folds) for training and validation over multiple iterations. That is, given a dataset $T = \{(x,y)\}$ it is divided into k disjoint subsets such that $T = \bigcup_{j=1}^{k} T_j$, where $T_j \bigcap T_{j'} = \varnothing \text{for} j \neq j'$, and each subset contains approximately the same number of data points (N/k), where N is the total number of data points. In each iteration, one subset T_j will be used as the validation set, while the remaining $(k-1)$ subsets will be used for training. Since the division used to train the ANN follows the previously mentioned method, the number of k-folds will be set to 7. Thus, 85 % of the data initially used in the neural network for training and validation will be distributed among the other supervised methods as 85.7 % for training and 14.3 % for cross-validation in each iteration.

In order to analyze the ability of the different classifiers to discriminate both the nominal class and the faults, confusion matrices will be used. The confusion matrix is row-normalized, allowing the percentages to be interpreted in terms of the proportion of correctly and incorrectly classified instances for each class. Each row corresponds to an actual class (True Class), and each column represents the predicted class (Predicted Class). The diagonal values indicate the number (or percentage) of correctly classified instances, while the off-diagonal values represent misclassifications. The metrics employed in the confusion matrix are shown in Equations (1)–(4). For a more detailed description of these metrics, see Larner [46].

Equation (1) shows the TPR (True Positive Rate), also known as sensitivity or recall. It measures the proportion of true positives (TP) among all actual positive instances (TP + FN), where FN are false negatives. This metric is useful for evaluating how well the model detects positive cases. A high TPR indicates that the model is highly sensitive to actual positives.

$$TPR = \frac{TP}{TP + FN} \tag{1}$$

Equation (2) shows the FNR (False Negative Rate). It measures the proportion of false negatives (FN) among all actual positive instances (TP + FN). It indicates the error rate where the model incorrectly classifies positive cases as negative. A low FNR is desirable to avoid false negatives.

$$FNR = \frac{FN}{TP + FN} \tag{2}$$

Equation (3) shows the PPV (Positive Predictive Value). Also known as precision. It measures the proportion of true positives (TP) among all positive predictions (TP + FP), where FP are false positives. It evaluates how reliable the model is when classifying an instance as positive. A PPV close to 1 indicates that most positive predictions are correct.

$$PPV = \frac{TP}{TP + FP} \tag{3}$$

Equation (4) Shows the FDR (False Discovery Rate). It measures the proportion of false positives (FP) among all positive predictions (TP + FP). It reflects the error rate when predicting positives that are actually not, where an FDR close to zero is desirable.

$$FDR = \frac{FP}{TP + FP} \tag{4}$$

4. Results

This section presents the main findings of the study, analyzing how they relate to the stated objectives. Additionally, a benchmarking analysis is conducted among the different alternatives, examining their limitations as well as their relevance for the detection of soft faults in analog circuits.

4.1. Classification using k-nearest neighbors (KNN)

The algorithm for this method is shown in Table 2. It is a supervised technique that classifies new observations based on their proximity to previously labeled points in the feature space. To apply this method, the "*fitcknn*" function from MATLAB™ 2022b [44] will be used, which allows training a KNN model by fitting it to a training dataset composed of predictor features and class labels. This model utilizes distance metrics such as Euclidean, Manhattan, or Mahalanobis, among others, to identify the k-nearest neighbors for each new observation [47,48]. One of the drawbacks of this method is that it requires storing all the training data, which can result in a high computational cost.

In the KNN method, the training set (x_i, y_i) for $i = 1..n$ where x_i represents the features and y_i corresponds to the class labels $(C_1..C_n)$, is stored as is, without performing an explicit learning process. To classify a new sample x_{new}, the algorithm computes the distance between x_{new} and each sample in the training set using a specific metric (such as Euclidean, Manhattan, or cosine). Then, the k-nearest neighbors are selected, meaning the k samples with the smallest distances. Finally, x_{new} is assigned to the class with the highest frequency among the k selected neighbors (majority voting). Some of the distances used in this method include Euclidean distance (5), Manhattan distance (6), Minkowski distance (7) (where p = 1 corresponds to Manhattan and p = 2 corresponds to Euclidean) and Mahalanobis distance (8), where S is the covariance matrix, among others. A detailed description of these metrics can be found in [47,48].

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{5}$$

$$d(x_i, x_j) = \sum_{k=1}^{n} |x_{ik} - x_{jk}| \tag{6}$$

$$d(x_i, x_j) = \left(\sum_{k=1}^{n} |x_{ik} - x_{jk}|^p \right)^{1/p} \tag{7}$$

$$d(x_i, x_j) = \sqrt{(x_i - x_j)^t S^{-1} (x_i - x_j)} \tag{8}$$

As was previously mentioned, the KNN model was optimized using the "*fitcknn*" function in MATLAB™, configuring the option '*OptimizeHyperparameters*', '*auto*' to automatically adjust the most relevant

Table 2
Summary of the KNN algorithm.

1	Compute the distances between x_{new} and all observations in the training set.
2	Identify the k nearest neighbors.
3	Assign weights (if necessary) based on distance.
4	Classify x_{new} according to the labels of the neighbors, using the weighted sum of votes.

hyperparameters: the number of neighbors (k) and the distance metric. Additionally, 7-fold cross-validation was used, as previously mentioned, to ensure the robustness of the results and reduce variance. The obtained results highlighted the Mahalanobis metric with 25 neighbors as the most effective configuration for this dataset, achieving a minimum average error of 0.023284. The confusion matrices shown in Figs. 5 and 6 represent the performance of a k-Nearest Neighbors model on the training and test data, respectively, configured with 25 neighbors and using the Mahalanobis distance as the metric. It can be observed that most classes have a 100 % correct classification rate, as seen in the first rows and columns. This indicates that the model performs well for these classes, correctly assigning all instances. However, for the R4high, R4low, R5high, and R5low classes, there is a higher number of errors, reflected in the values outside the diagonal. This could indicate that these two classes share similar characteristics, making their differentiation more challenging with the current model.

Regarding the most challenging classes, the model demonstrated lower accuracy. For example, as shown in Fig. 5, for the R4high class, the model correctly identifies 92.6 % of the instances (TPR) and has a precision of 87.7 % (PPV). The false negative rate, where instances are misclassified as R5low, is 7.4 % (FNR), and the false discovery rate is 12.3 % (FDR). Similarly, for R4low class, the model correctly identifies 90.6 % of the instances (TPR) and has a precision of 92.3 % (PPV). The false negative rate, where instances are incorrectly classified as R5high, is 9.4 % (FNR), while the false discovery rate, where they are also classified as R5high, remains at 7.7 % (FDR). Regarding the test set, the confusion matrix in Fig. 6 follows a trend similar to that of the training set, shown in Fig. 5, where it can be observed that, for the most problematic classes, TPR and PPV percentages decrease, demonstrating lower performance.

As observed in Fig. 5, the R4high and R5low classes are misclassified when detecting soft faults, and the same occurs with the R4low and R5high classes. Regarding the test set, Fig. 6 reveals a TPR of 100.0 % for the R5high class, indicating that the model correctly identified all instances. Nevertheless, precision decreases to 78.6 % (PPV), suggesting an increase in the proportion of false positives. Additionally, the false discovery rate (FDR) increases to 21.4 %, indicating less consistent performance in terms of precision outside the training set. However, since the rest of the fault classes, as well as the nominal values, were correctly identified, it could be stated that the KNN model performs well across all classes, except for the mentioned cases. Moreover, overall, the TPR and PPV rates remain high.

4.2. Analysis of results for Naïve Bayes (NB)

The Naïve Bayes (NB) algorithm, whose procedure is shown in Table 3, is a probabilistic model based on Bayes' theorem (9).

$$P(C/X) = \frac{P(X/C)P(C)}{P(X)} \tag{9}$$

Where $P(C/X)$ is the probability that X belongs to class C (nominal value or soft faults), $P(X/C)$ is the conditional probability indicating how likely it is to observe feature X given that it belongs to class C, and $P(C)$ is the probability of class C, based solely on the frequency of that class in the data. $P(X)$ represents the total probability of observing X, regardless of the class. That is, it is the sum $P(X) = \sum_C P(X/C)P(C)$ for all possible classes. For a more detailed description of the method, see Refs. [47,48].

The objective of the algorithm is to determine the class C that maximizes $P(C/X)$, that is, the probability that an observation X belongs to a specific fault class. In this case, the Naïve Bayes algorithm assumes conditional independence between features, which implies that each feature contributes independently to the probability of the class.

In this study, the Naïve Bayes model was optimized to classify a dataset using the same training data. Through a Bayesian optimization process, Gaussian (normal) distributions and kernel-based distributions were explored using the *"fitcnb"* function in MATLAB™ 2022b [44]. The final model selected normal distributions for all features, achieving a minimum cross-validation loss of 0.1380. As shown in the confusion matrices in Figs. 7 and 8, obtained with this method, the model is unable to accurately predict either the soft faults or nominal values.

Figs. 7 and 8 show the confusion matrices of the Gaussian Naïve Bayes model for the training and test sets, respectively. As observed, in both the training and test sets, the model demonstrates poor overall performance compared to KNN, making it unsuitable for detecting soft faults in this type of analog electronic circuit.

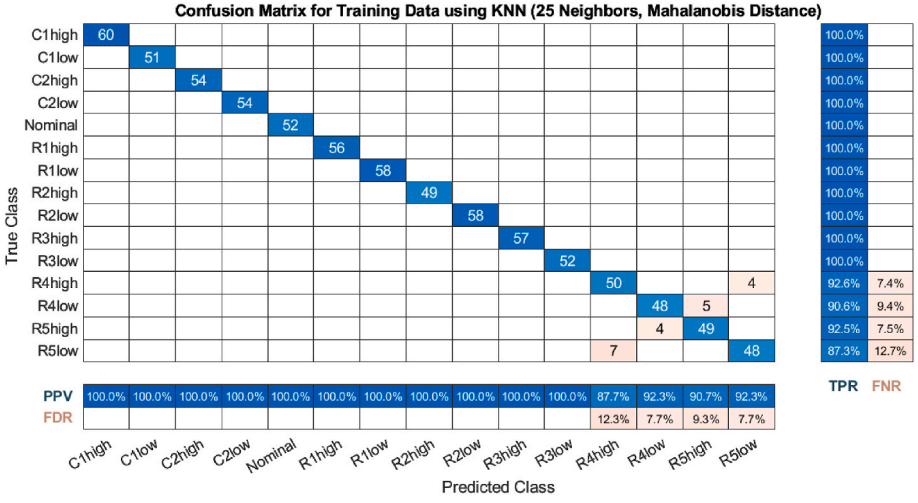

Fig. 5. Confusion matrix for the training data (selected KNN).

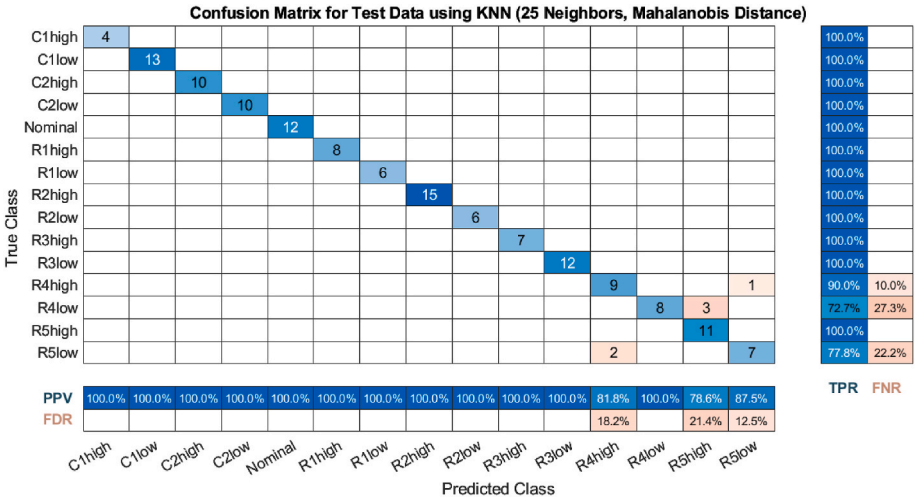

Fig. 6. Confusion matrix for the test data (selected KNN).

Table 3
Summary of the Naïve Bayes (NB) algorithm.

1 Training: Compute the probability of each class $P(C)$.
2 Compute $P(x_i/C)$, assuming independence.
3 For an observation $X = \{x_1 \ldots x_6\}$, compute $P(C/X)$.
4 Assign the class with the highest probability $P(C/X)$.

4.3. Analysis of results for Discriminant Analysis Classifier (DAC)

The discriminant analysis method implemented in the *"fitcdiscr"* function of Matlab™2022b [44] is an implementation of Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA), specifically designed for classification problems as shown by Table 4. This method is based on finding a linear or quadratic combination of the predictor variables (features) that maximizes the separation between the target classes [47,48].

Fig. 9 shows the confusion matrix of the Discriminant Analysis Classifier (DAC) in the training set. Overall, the model demonstrates good performance, with correct predictions for most classes. The performance of the DAC model is similar to that of KNN, with R4high, R4low, R5high, and R5low being the most challenging classes to identify. However, in the case of DAC, one instance belonging to the nominal class is misclassified as R2high, resulting in slightly lower performance in this case.

Fig. 10 presents the confusion matrix of DAC in the test set. Classes such as C1low, Nominal, and R1high maintain excellent performance, with a TPR of 100 %, indicating that all actual instances of these classes were correctly classified. However, other classes, such as R4low, R4high, R5low, and R5high, exhibit lower TPR values compared to the training set.

In summary, the Discriminant Analysis Classifier model shows good overall performance, particularly in the training set, where it achieves high correct prediction rates for most classes. However, in the test set, some classes, such as R4low, R4high, R5low, and R5high, exhibit generalization issues similar to those observed in KNN. Despite misclassifying one instance from the nominal class as R2high, no fault instances were incorrectly classified as nominal using this method, which, if it had occurred, would have been more problematic.

4.4. Classification Decision Tree (CDT)

In this case, a decision tree-based classification model is trained using the *"fitctree"* function in Matlab™2022b [44] as shown in Table 5, automatically optimizing the most relevant model hyperparameters through Bayesian optimization. The predictive data (x) and class labels (y) serve as inputs to train the tree, and the hyperparameter search adjusts the minimum leaf size.

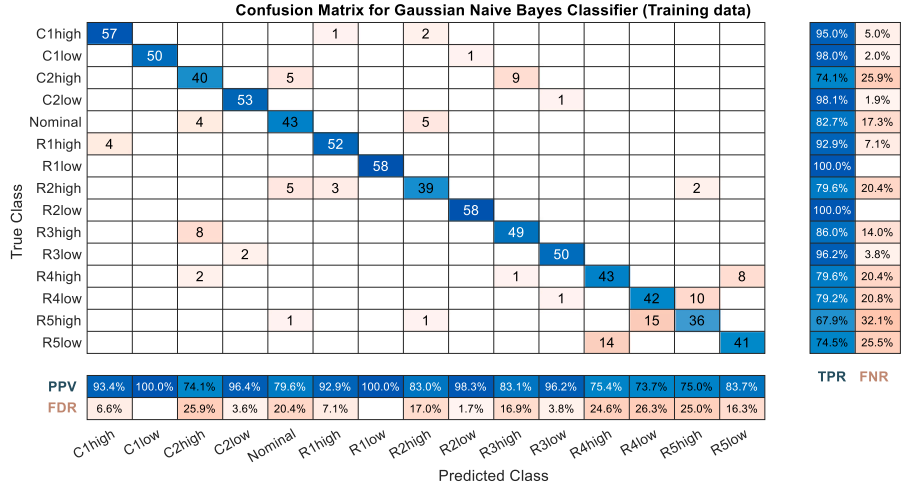

Fig. 7. Confusion matrix for the training data (Gaussian Naïve Bayes).

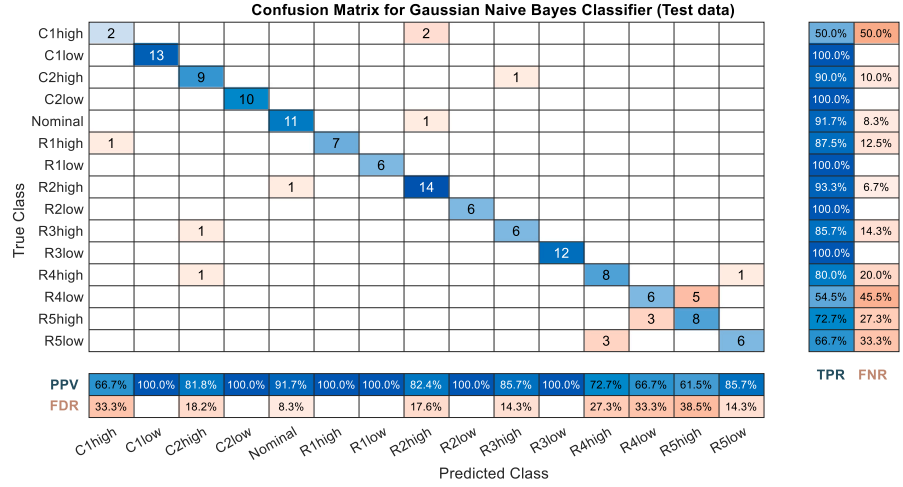

Fig. 8. Confusion matrix for the test data (Gaussian Naïve Bayes).

Table 4
Summary of the DAC algorithm using the *"fitcdiscr"* function of Matlab™2022b [44].

1. Specify the number of k-folds and max iterations.
2. Select automatic hyperparameter optimization.
3. Use Bayesian optimization to find the best model configuration.
4. The *"fitcdiscr"* function tests different hyperparameter combinations and trains a discriminant analysis model.

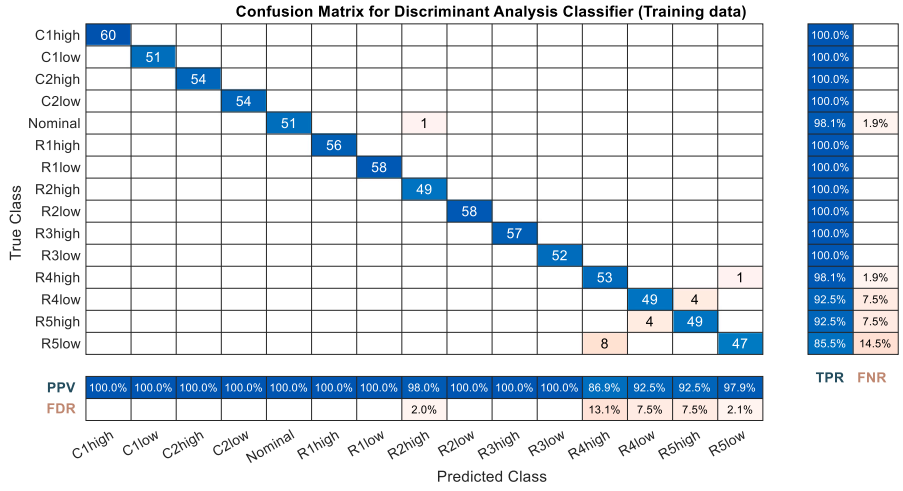

Fig. 9. Confusion matrix for the training data (Discriminant Analysis Classifier).

As observed in the confusion matrices in Figs. 11 and 12, for training data and test data, respectively, the decision tree exhibits irregular performance in both the training and test sets. Additionally, the model incorrectly predicts fault values as belonging to the nominal class, which is more problematic. Specifically, one instance of C2high, two of R2high, and one of R5high are incorrectly predicted as nominal. In the test set, shown in Fig. 12, the classification accuracy for different classes improves compared to the training set, but one instance of R2high is still misclassified as nominal.

4.5. Random forest

Random Forest is a supervised algorithm that combines decision trees trained with random data and variables, improving accuracy and reducing overfitting [47]. In this study, a classification model based on an ensemble of decision trees is trained using the Bagging (*Bootstrap Aggregating*) method with the *"fitcensemble"* function in Matlab™ 2022b [44]. The model training is automatically optimized through hyperparameter tuning, using a Bayesian optimization process. The hyperparameter search employs an acquisition function called *"expected-improvement-plus"*. Additionally, the model evaluation during the optimization process is conducted through cross-validation, ensuring that the performance metrics are representative and not biased by a specific training dataset.

The graph depicted in Fig. 13 confirms that the optimization successfully identified an efficient model in terms of the objective function (minimum around 0.080882). Additionally, the convergence of the observed and estimated values supports the quality of the Bayesian optimization model.

As observed in Fig. 14, the Random Forest model provides significant

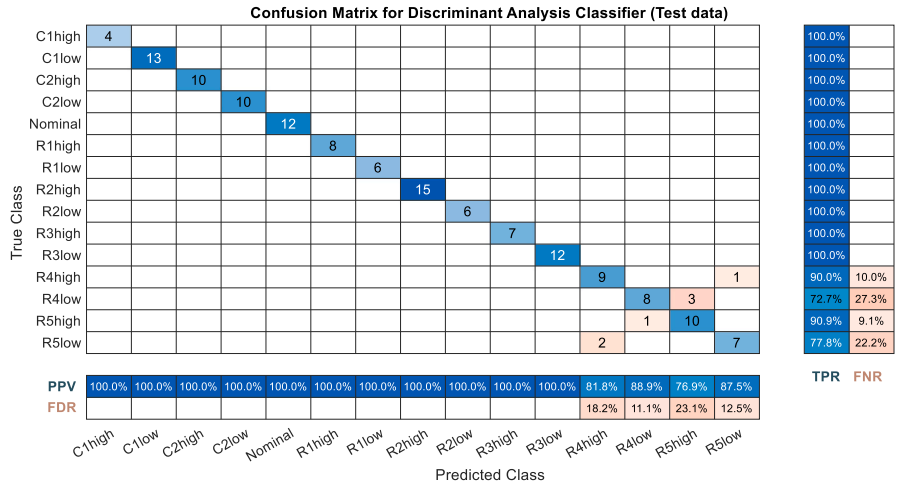

Fig. 10. Confusion matrix for the test data (Discriminant Analysis Classifier).

Table 5
Summary of the CDT Algorithm the *"fitctree"* function in Matlab™2022b [44].

1.Specify the number of k-folds and the maximum number of objective function evaluations.
2.Enable automatic hyperparameter optimization ('OptimizeHyperparameters', 'auto').
3.Apply Bayesian optimization, using the *"expected-improvement-plus"* acquisition function, to find the best model configuration.
4.The *"fitctree"* function tests different hyperparameter combinations (such as tree depth and splitting criterion) and adjusts a decision tree classification model according.

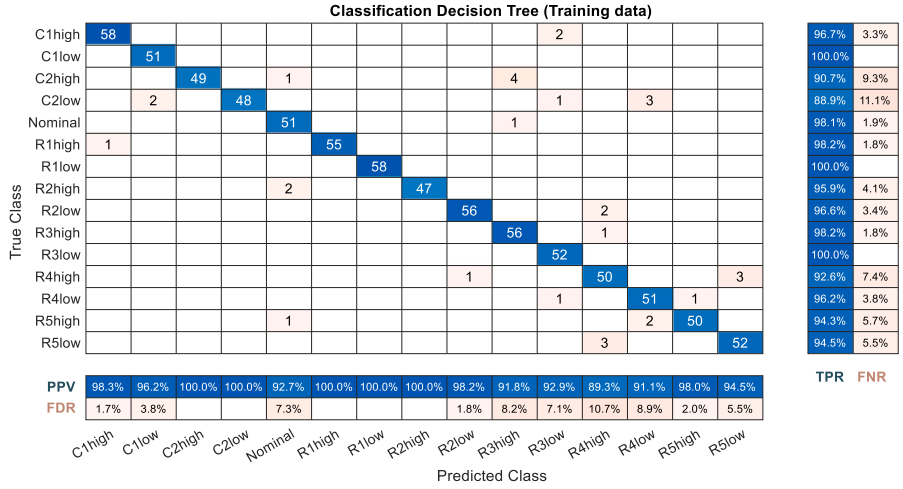

Fig. 11. Confusion matrix for the training data (Classification Decision Tree).

improvements over the KNN and CDT classifiers in terms of performance on test data. However, it has the drawback of misclassifying a C2high instance as nominal, which could be problematic. Regarding the values used for testing, the classification matrix in Fig. 15 shows that the model's performance deteriorates, compared to the training data, in the R4high, R4low, R5high, and R5low classes. However, for the remaining classes, the TPR reaches 100 % true positive rate.

4.6. Support Vector Machines (SVM)

Another machine learning technique used in this study is Support Vector Machines. Specifically, in this study, the *"fitcecoc"* function in Matlab™2022b [44] was used, which trains a multiclass classifier using Error-Correcting Output Codes, decomposing the problem into multiple binary subproblems solved with Support Vector Machines (SVM). A linear kernel is used to separate the classes in the original feature space. The model automatically optimizes its hyperparameters through cross-validation and Bayesian optimization. As is well known, the supervised learning method called Support Vector Machines (SVM) finds an optimal hyperplane to separate classes in a feature space. To achieve this, it utilizes support vectors and can employ kernel functions [48].

Fig. 16 presents the results obtained, which shows the evolution of the objective function with the number of iterations. It can be observed that it rapidly decreases until reaching 9–10 iterations, after which it stabilizes once the optimum of 0.02384 is achieved.

The SVM model shows good performance in both the training and test data. In the training data, shown in Fig. 17, the TPR (True Positive Rate) is close to 100 % for most classes. However, there are slight decreases in classes such as R4high, R4low, R5high, and R5low, where the TPR ranges between 92.5 % and 98.1 %. Fig. 18 presents the confusion

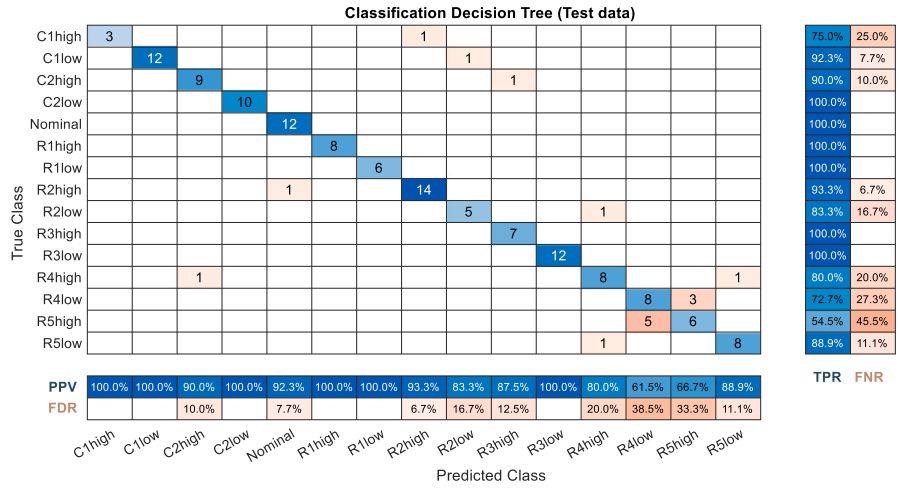

Fig. 12. Confusion matrix for the test data (Classification Decision Tree).

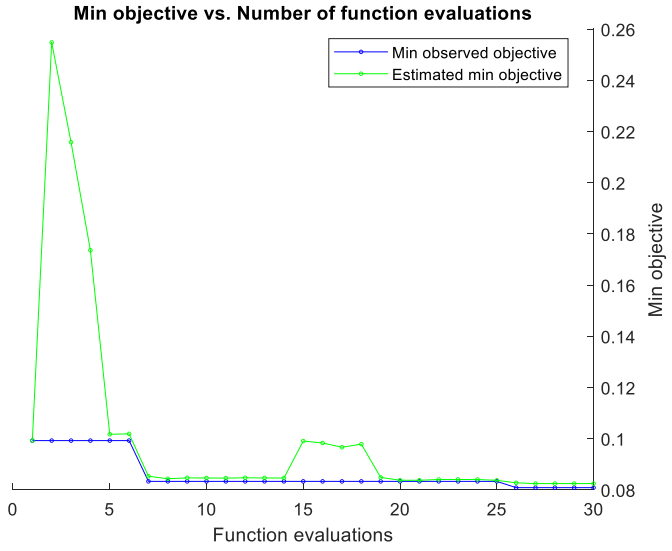

Fig. 13. (a) evolution of the objective function using RF

matrix for the test data, where SVM maintains good performance, with a TPR of 100 % for most classes. Although its performance decreases slightly compared to the training data in the R4low, R5high, and R5low classes, where the TPR reaches 81.8 %, 90.9 %, and 88.9 %, respectively, this confirms that these classes remain challenging to detect, consistent with the findings in the other analyzed models.

4.7. *Artificial neural network*

Finally, a Patternet-type neural network [45] is employed, as shown in Fig. 19, where X is the input vector (6×1), W_{hidden} (8x6) is the weight matrix of the hidden layer, connecting the six inputs to the 8 hidden neurons, b_{hidden} (8x1) is the bias vector of the hidden layer, introducing an additional offset to the linear combinations of the inputs, z_{hidden} (8x1) is the weighted input vector (pre-activations) for the hidden layer, a_{hidden} (8x1) is the hidden layer output after applying the activation function and f_{hidden} is the activation function used, which is sigmoidal. Additionally, W_{output} (15x8) is the weight matrix of the output layer, connecting the 8 hidden neurons to the 15 output neurons, b_{output} is the bias vector of the output layer, z_{output} (15x1) is the weighted input vector (pre-activations) for the output layer, f_{output} is the activation function of the output layer, which is of type softmax, z_i represents the pre-activation of the $i-th$ output neuron and a_{output} (15x1) is the final output of the network.

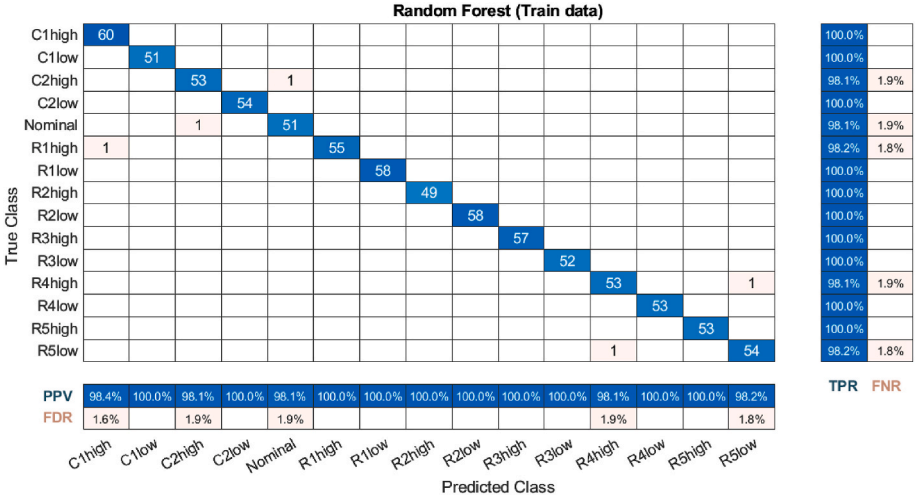

Fig. 14. Confusion matrix for the training data (Random Forest).

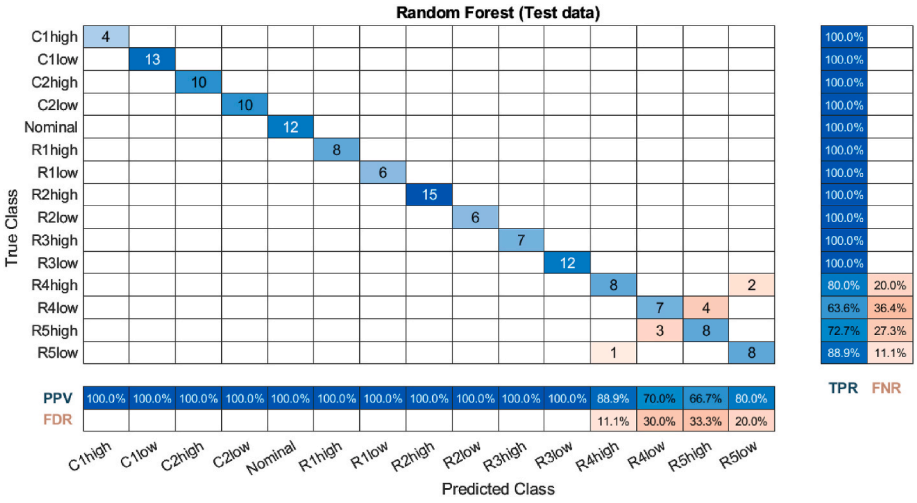

Fig. 15. Confusion matrix for the test data (Random Forest).

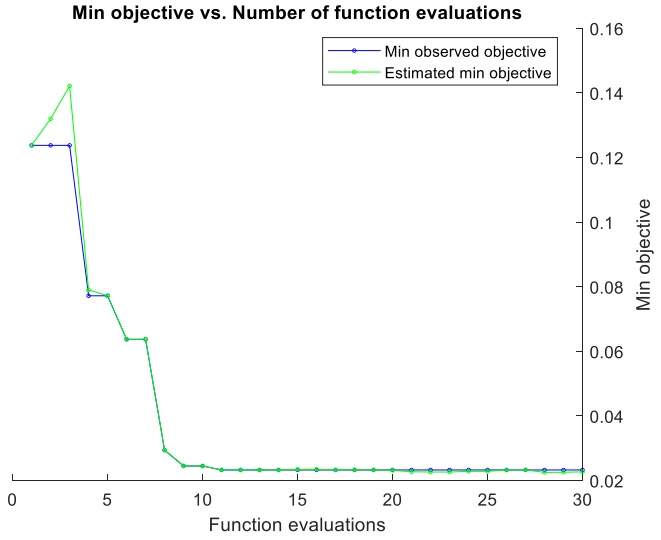

Fig. 16. Evolution of the objective function.

As observed in Fig. 19, a sigmoidal transfer function is applied in the hidden layer. The network employs a feedforward propagation approach to compute the outputs. As mentioned in the methodology section, the process begins with the random division of data (*dividerand*) into training, validation, and test sets, preventing bias in the model. These same datasets were used in the previously analyzed classifiers. Training is performed using the scaled conjugate gradient algorithm, implemented through the Matlab™ function (*trainscg*), which optimizes the weights iteratively without requiring the computation of the Hessian matrix. Additionally, the selected loss function is cross-entropy, which measures the discrepancy between the probabilities predicted by the model and the actual classes.

Fig. 20(a) shows the evolution of cross-entropy during the training, validation, and testing of the neural network, highlighting the best performance in validation. On the other hand, Fig. 20(b) presents an error histogram, displaying the distribution of differences between the network's predictions and the actual labels in the training, validation, and test sets. Most errors are concentrated near zero, indicating a good overall fit of the model.

As can be observed in Fig. 21, the confusion matrix for the ANN presents an almost perfect performance with the training data, with TPR (True Positive Rate) values close to 100 % for most classes. This indicates that the neural network has learned to classify the training data correctly with very few errors. However, there are some exceptions,

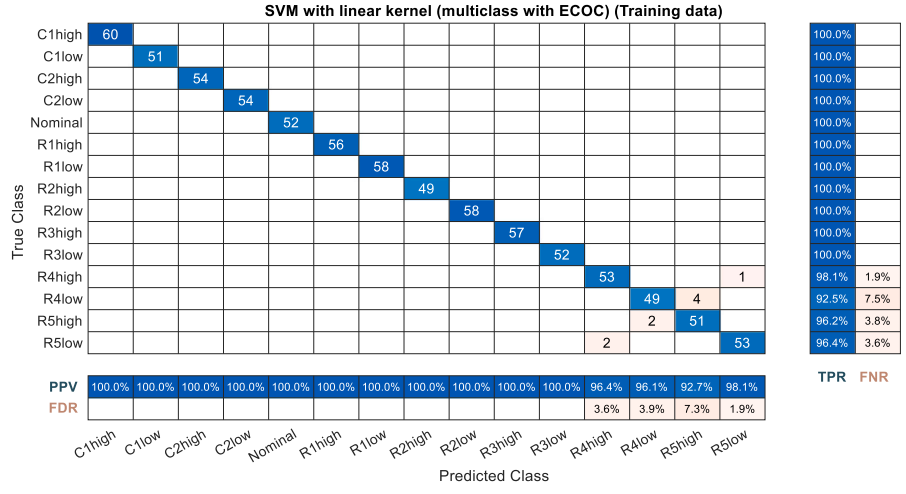

Fig. 17. Confusion matrix for SVM (training data).

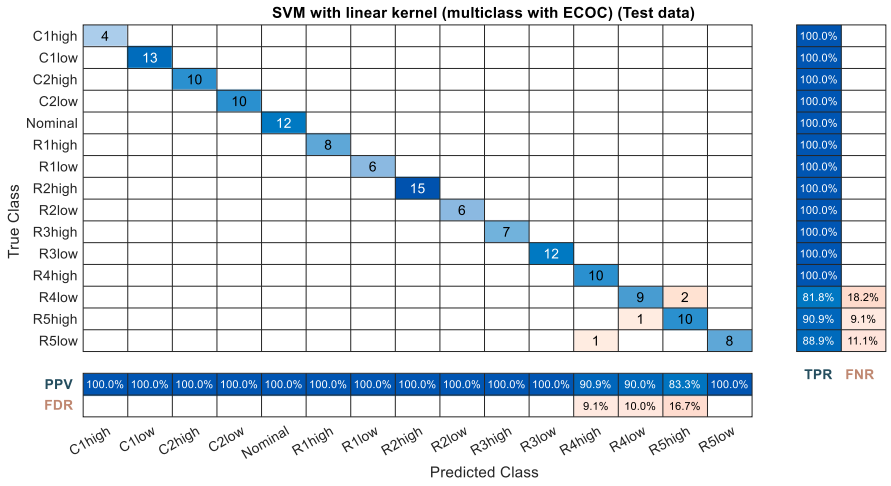

Fig. 18. Confusion matrix for SVM (test data).

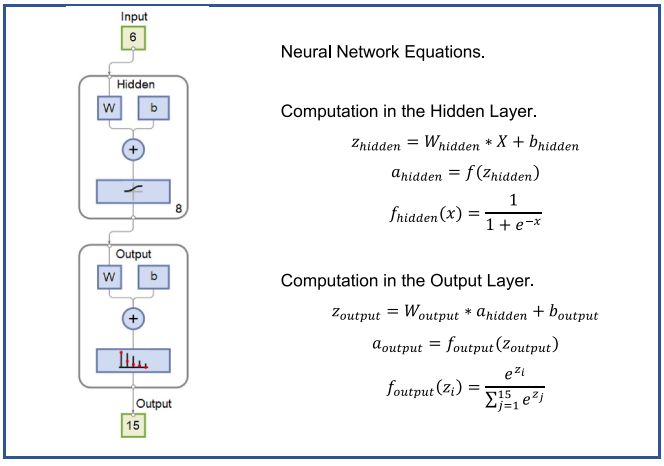

Fig. 19. Neural network employed and equations.

such as the R4low, R5high, and R5low classes, where the TPR slightly decreases to 96.2 %, 96.4 %, and 94.3 %, respectively. Nevertheless, these values are lower than those obtained previously with the other classifiers analyzed in this study. On the other hand, in the confusion matrix obtained with the test data, which is shown in Fig. 22, it can be observed that the results follow a similar trend to those of the training set, correctly classifying most classes.

In summary, the ANN is highly effective for this classification problem, with very few discrepancies between predictions and actual labels, even in more complex classes, showing superior performance compared to the other classifiers analyzed in this section, as will be demonstrated in the following section.

5. Discussion of results

To compare the models as a whole, the metrics shown in Equations (10)–(15) are used. The global accuracy of each classifier is given by Equation (10), which represents the percentage of correct predictions made by the model out of the total instances evaluated, being N the number of data points.

$$Global\ accuracy_{Classifier} = \frac{Number\ of\ correct\ predictions_{Classifier}}{N} \tag{10}$$

Global precision, given by Equation (11), is the average precision of each class.

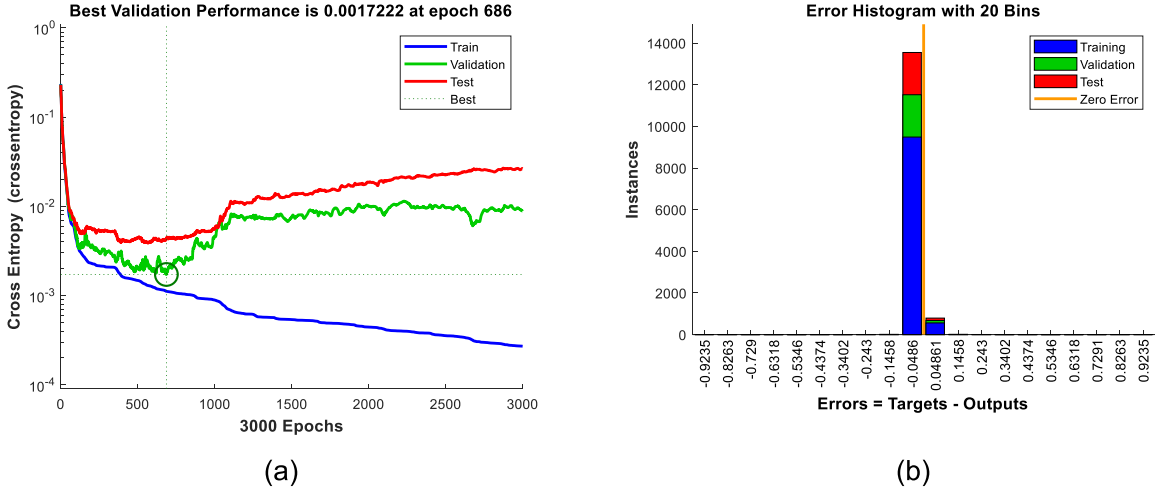

Fig. 20. Neural network employed: (a) Evolution of cross-entropy during the training, validation, and testing of the neural network and (b) Error histogram.

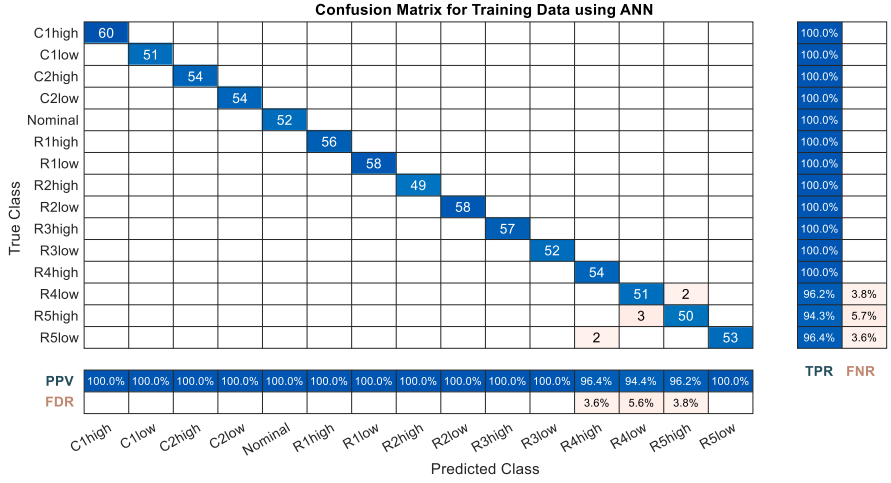

Fig. 21. Confusion matrix for the training data (ANN).

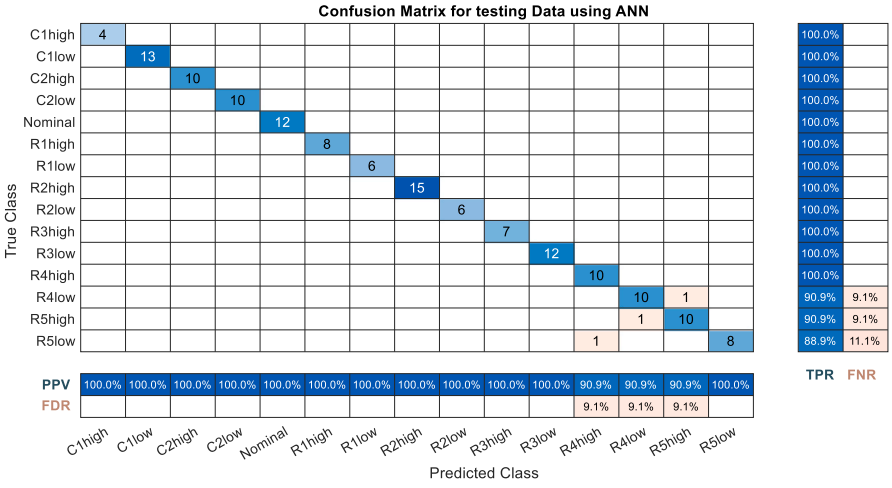

Fig. 22. Confusion matrix for the test data (ANN).

$$Global\ precision_{Classifier} = \frac{\sum_{i=1}^{Num.\ class} Precision_i}{N} \tag{11}$$

Global recall, given by Equation (12), is the average recall of each class.

$$Global\ recall_{Classifier} = \frac{\sum_{i=1}^{Num.\ class} Recall_i}{N} \tag{12}$$

$F1 - score$ represents the harmonic mean between precision and recall for each class and is evaluated from Equation (13).

$$F1 - score = 2^{*}\frac{Precision^{*}Recall}{Precision + Recall} \tag{13}$$

And the *Global* $F1 - score$ is given by Equation (14), which is evaluated for each classifier.

$$Global\ F1 - score_{Classifier} = \frac{\sum_{i=1}^{Num.\ class} F1 - score_i}{N} \tag{14}$$

Finally, Equation (15) shows the Matthews Correlation Coefficient (MCC) [49], which is a metric used in classification problems to measure the quality of a model, considering all elements of the confusion matrix. Its value ranges between −1 and + 1, where +1 indicates perfect classification, 0 represents random performance, and −1 means completely incorrect classification [50–52].

$$MCC = \frac{TP^{*}TN - FP^{*}FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{15}$$

Tables 6 and 7 show the global results obtained with the machine learning techniques analyzed for the training and test data, respectively.

As shown in Table 6, with the training data, Random Forest (RF) is the best-performing classifier across all metrics, achieving 99.39 % in accuracy, precision, and F1-score, 99.38 % in recall, and an MCC of 99.34 %. This indicates that RF has an almost perfect predictive capability on the training data. Similarly, although the Artificial Neural Network (ANN) performs at a very high level, it is slightly below RF. On the other hand, Support Vector Machine (SVM) ranks third. However, its performance is slightly lower than that of ANN, but it remains a suitable model for fault classification. Likewise, the Discriminant Analysis Classifier (DAC) also demonstrates good performance. Meanwhile, the Classification Decision Tree (CDT) exhibits a decline in performance, making it less effective compared to the previously mentioned models. Finally, Naïve Bayes (NB) is the worst-performing classifier across all metrics, obtaining 87.13 % accuracy, 86.98 % precision, 86.92 % recall,

Table 6
Performance comparison between classifiers (training data).

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	MCC (%)
KNN	97.55	97.54	97.53	97.52	97.38
NB	87.13	86.98	86.92	86.90	86.22
DAC	97.79	97.85	97.77	97.76	97.64
CDT	96.08	96.21	96.07	96.06	95.81
RF	99.39	99.39	99.38	99.39	99.34
SVM	98.90	98.89	98.88	98.88	98.82
ANN	99.14	99.14	99.13	99.13	99.08

Table 7
Performance comparison between classifiers (test data).

	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	MCC (%)
KNN	95.83	96.53	96.03	96.02	95.56
NB	86.11	86.32	84.81	85.26	85.05
DAC	95.14	95.68	95.43	95.43	94.78
CDT	88.89	89.57	88.68	88.88	88.04
RF	93.06	93.70	93.68	93.64	92.52
SVM	97.22	97.62	97.44	97.47	97.01
ANN	97.92	98.18	98.05	98.08	97.76

86.90 % F1-score, and an MCC of 86.22 %. This demonstrates that NB is the least efficient model in this comparison, likely due to the assumption of independence among features, which may not hold in this dataset.

Although in advance it seems that RF is the best technique for fault detection in the CUT, when validating its performance on the test data, a significant decrease is observed. Specifically, as shown in Table 7, the Artificial Neural Network (ANN) is the best-performing classifier on the test data, achieving 97.92 % accuracy, 98.18 % precision, 98.05 % recall, 98.08 % F1-score, and an MCC of 97.76 %. This indicates that ANN maintains a high level of performance and generalization, with only a slight reduction compared to the training data. The Support Vector Machine (SVM) ranks second in performance. Its performance is very close to that of ANN, demonstrating a strong classification capability on the test data. On the other hand, k-Nearest Neighbors (KNN) also demonstrates good performance. The Discriminant Analysis Classifier (DAC) shows a performance similar to that of KNN, although with a slight decrease compared to the training data. However, contrary to what was observed with the training data, Random Forest (RF) achieved lower performance on the test data compared to its training performance, with 93.06 % accuracy, 93.70 % precision, 93.68 % recall, 93.64 % F1-score, and an MCC of 92.52 %. This drop in performance suggests the possibility of some overfitting in the model. Similarly, the Classification Decision Tree (CDT) exhibits lower performance compared to the other models. Finally, Naïve Bayes (NB) is the lowest-performing classifier. Its lower recall value indicates that it struggles to detect certain classes, which may be likely caused by the limitations of its independence assumption among features.

Therefore, based on the data obtained in Tables 6 and 7 and it is observed that Artificial Neural Network (ANN) and Support Vector Machine (SVM) are the best-performing models overall. ANN achieves the highest values across all metrics on the test data (97.92 % accuracy and an MCC of 97.76 %), with a slight reduction compared to training, suggesting good generalization capability. SVM closely follows with 97.22 % accuracy and an MCC of 97.01 %, also demonstrating high performance on both datasets. At a second performance level, k-Nearest Neighbors (KNN) and Discriminant Analysis Classifier (DAC) exhibit similar values in both the training and test datasets. KNN drops from 97.55 % accuracy in training to 95.83 % in testing, with its MCC decreasing from 97.38 % to 95.56 %, while DAC shows a decline from 97.79 % to 95.14 % in accuracy and from 97.64 % to 94.78 % in MCC.

Random Forest (RF) shows irregular behavior: although it achieves the best performance in training (99.39 % across all metrics), its performance drops significantly in testing (93.06 % accuracy and an MCC of 92.52 %). This indicates possible overfitting, as its test performance is not competitive compared to models like ANN and SVM. At the lower performance levels, Classification Decision Tree (CDT) and Naïve Bayes (NB) are found. CDT drops in accuracy from 96.08 % in training to 88.89 % in testing, with an MCC of 88.04 %, indicating a clear performance loss on unseen data. Meanwhile, Naïve Bayes (NB) is consistently the worst classifier, with an accuracy of 87.13 % in training and 86.11 % in testing, and the lowest MCC in both phases (86.22 % in training and 85.05 % in testing), confirming its lower predictive capacity in this context. In conclusion, ANN and SVM are the most reliable and best-performing models overall, followed by KNN and DAC, which exhibit competitive performance. Random Forest, despite its high training performance, shows signs of overfitting, while CDT and NB are the least effective classifiers in this dataset.

On the other hand, Figs. 23 and 24 graphically present the results obtained in Tables 6 and 7, for the training and test data, respectively. As observed in Fig. 23, the RF method achieves the best results with the training data. However, its performance decreases more significantly with the test data, as shown in Fig. 24.

Therefore, it is evident that ANN is superior to the other classification models used, as it achieves a better balance between training and test data than the rest. Additionally, as shown in the results section, RF, although it achieved better performance in the training data, had the drawback of misclassifying a C2high instance as nominal, which could be problematic. Likewise, SVM, while producing slightly lower results than ANN, is a solid alternative for predicting soft faults in analog circuits. On the other hand, KNN is another method capable of effectively distinguishing the nominal class from the faulty classes, although its accuracy rates in the other classes are lower.

Fig. 25 shows that incorporating the three most influential variables, selected from those with the highest variance and correlation, improves class separation. However, it is necessary to increase the number of independent variables in the classifiers to achieve better class separation. The figure also shows that SVM and RF define the class separation boundaries more clearly.

Finally, Fig. 26 shows the classification performed with the ANN using the entire dataset. It can be observed that there is better class separation, which aligns with the data obtained in Tables 6 and 7 (for training and test, respectively). This suggests that the ANN is better able to model the complexity of these data, as can be seen from Fig. 26 and the previously obtained metrics.

6. Conclusions

In this study, various supervised classification techniques have been evaluated for the detection and classification of soft faults in analog circuits. A feature extraction method based on voltage measurements at key points in the circuit and at three different frequencies was used, allowing for the extraction of relevant information for fault diagnosis in the Sallen-Key band-pass filter, considered as the circuit under test in this study.

The results show that Artificial Neural Network (ANN) and Support Vector Machines (SVM) are the most effective classifiers for fault detection in the CUT, achieving 97.92 % and 97.22 % accuracy on the test data, with MCC values of 97.76 % and 97.01 %, respectively. ANN demonstrated better generalization, maintaining a minimal difference between its training and test performance. In contrast, Random Forest (RF) achieved the best performance in training (99.39 % accuracy, MCC of 99.34 %) but suffered a notable drop in testing (93.06 % accuracy, MCC of 92.52 %), also misclassifying faulty classes as nominal, which is more problematic.

Models such as KNN (95.83 % accuracy, MCC of 95.56 %) and Discriminant Analysis Classifier (DAC) (95.14 % accuracy, MCC of 94.78 %) demonstrated solid performance, although inferior to ANN and SVM. At the opposite end, Classification Decision Tree (CDT) and Naïve Bayes (NB) were the worst-performing classifiers, with significant

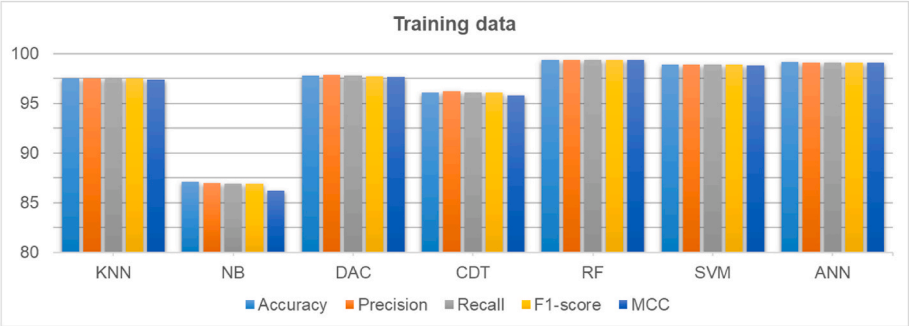

Fig. 23. Performance Comparison of Prediction Methods with training data.

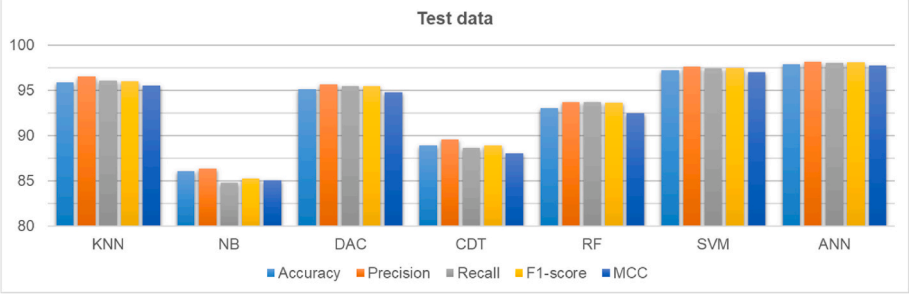

Fig. 24. Performance Comparison of Prediction Methods with test data.

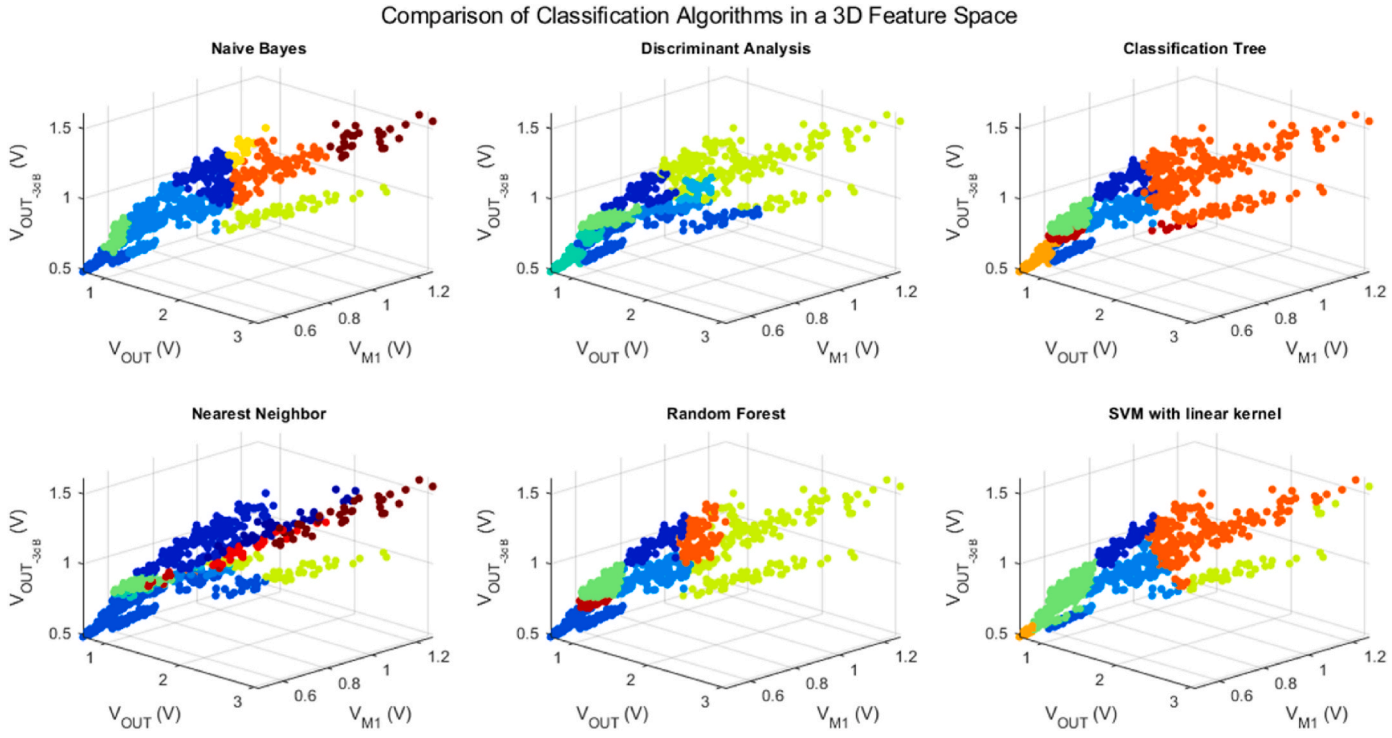

Fig. 25. Comparison of class separation across different classifiers when incorporating the three most influential variables.

reductions in their accuracy when evaluated on test data.

The analysis of decision regions revealed that Naïve Bayes and Discriminant Analysis exhibit high class overlap, making fault identification in the CUT more challenging. In contrast, Random Forest and SVM achieved more detailed and adaptive decision boundaries, although class separation remains limited when few variables are used. By incorporating the three most influential variables, a significant improvement in classification was observed, especially with SVM and RF. However, it was necessary to include six predictive variables, obtained from measurements at two circuit points at three different frequencies, to improve the separation between faulty and nominal classes.

Furthermore, the proposed feature extraction method has proven effective when combined with these classifiers for soft fault detection in the CUT. The ability of this approach to capture relevant information has

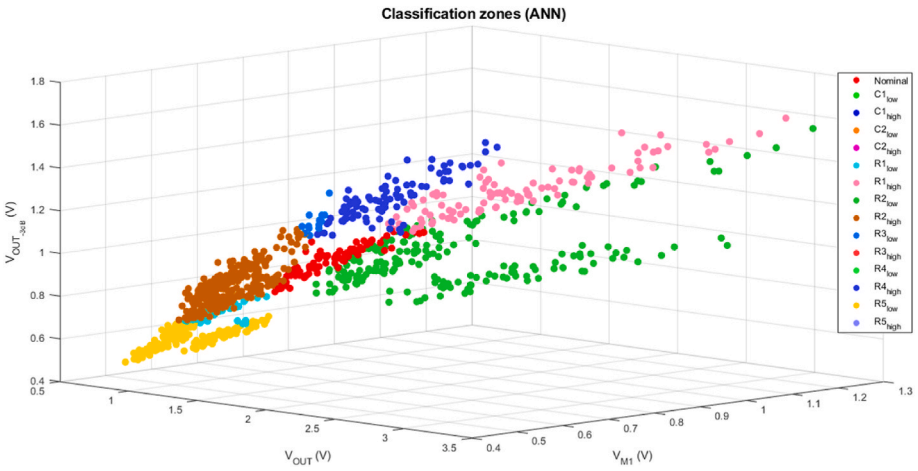

Fig. 26. Comparison of class separation in the ANN when incorporating the three most influential variables.

been key to the models' performance, highlighting its usefulness in electronic circuit diagnostics.

Future research will focus on extending the proposed methodology to other analog circuit topologies, as well as on its application to the identification of multiple faults and incipient faults in analog electronic circuits, and on experimental validation.

Funding statement

Open access funding provided by Universidad de Burgos (Spain).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The author acknowledges open access granted by Universidad de Burgos (Spain).

Data availability

The data used in this manuscript will be provided upon reasonable request.

References

[1] J. Qu, X. Fang, Y. Chai, Q. Tang, J. Liu, An intermittent fault diagnosis method of analog circuits based on variational modal decomposition and adaptive dynamic density peak clustering, Soft Comput. 26 (2022) 8603–8615, https://doi.org/10.1007/s00500-022-07226-1.

[2] E. Afacan, N. Lourenço, R. Martins, G. Dündar, Review: machine learning techniques in analog/RF integrated circuit design, synthesis, layout, and test, Integration 77 (2021) 113–130, https://doi.org/10.1016/j.vlsi.2020.11.006.

[3] X. Fang, J. Qu, Y. Chai, Self-supervised intermittent fault detection for analog circuits guided by prior knowledge, Reliab. Eng. Syst. Saf. 233 (2023) 109108, https://doi.org/10.1016/j.ress.2023.109108.

[4] S. Wang, Z. Liu, Z. Jia, Z. Li, Incipient fault diagnosis of analog circuit with ensemble HKELM based on fused multi-channel and multi-scale features, Eng. Appl. Artif. Intell. 117 (2023) 105633, https://doi.org/10.1016/j.engappai.2022.105633.

[5] X. Tang, A. Xu, Multi-class classification using kernel density estimation on K-nearest neighbours, Electron. Lett. 52 (2016) 600–602, https://doi.org/10.1049/el.2015.4437.

[6] J. Sun, G. Hu, C. Wang, Analog circuit soft fault diagnosis based on sparse random projections and K-Nearest neighbor, Sci. Program. 2021 (2021), https://doi.org/10.1155/2021/8040140.

[7] A. Deb, A.K. Jain, An effective data-driven machine learning hybrid approach for fault detection and classification in a standalone low-voltage DC microgrid, Electr. Eng. 106 (2024) 6199–6212, https://doi.org/10.1007/s00202-024-02334-7.

[8] A. Zare, M. Simab, M. Nafar, E.M.G. Rodrigues, A Novel Method for Fault Diagnosis in Photovoltaic Arrays Used in Distribution Power Systems, Springer Berlin Heidelberg, 2024, https://doi.org/10.1007/s12667-024-00706-3.

[9] S.R. Madeti, S.N. Singh, Modeling of PV system based on experimental data for fault detection using kNN method, Sol. Energy 173 (2018) 139–151, https://doi.org/10.1016/j.solener.2018.07.038.

[10] S. Awasthi, G. Singh, N. Ahamad, Classifying electrical faults in a distribution System using K-Nearest neighbor (KNN) model in presence of multiple distributed generators, J. Inst. Eng. Ser. B. 105 (2024) 621–634, https://doi.org/10.1007/s40031-024-00994-4.

[11] S. Roy, S.K. Millican, V.D. Agrawal, A Survey and recent advances: machine intelligence in electronic testing, J. Electron. Test. Theory Appl. 40 (2024) 139–158, https://doi.org/10.1007/s10836-024-06117-7.

[12] T. Zhang, T. Li, Analog circuit soft fault diagnosis utilizing matrix perturbation analysis, Analog Integr, Circuits Signal Process. 100 (2019) 181–192, https://doi.org/10.1007/s10470-019-01433-x.

[13] J. Shi, Y. Deng, Z. Wang, Analog circuit fault diagnosis based on density peaks clustering and dynamic weight probabilistic neural network, Neurocomputing 407 (2020) 354–365, https://doi.org/10.1016/j.neucom.2020.04.113.

[14] J. Shi, Y. Deng, Z. Wang, X. Guo, An adaptive new state recognition method based on density peak clustering and voting probabilistic neural network, Appl. Soft Comput. J. 97 (2020) 106835, https://doi.org/10.1016/j.asoc.2020.106835.

[15] B. Sudha, L.S. Praveen, A. Vadde, Classification of faults in distribution transformer using machine learning, Mater. Today Proc. 58 (2022) 616–622, https://doi.org/10.1016/j.matpr.2022.04.514.

[16] W. He, Y. He, B. Li, C. Zhang, A naive-bayes-based fault diagnosis approach for analog circuit by using image-oriented feature extraction and selection technique, IEEE Access 8 (2020) 5065–5079, https://doi.org/10.1109/ACCESS.2018.2888950.

[17] A. Arabi, M. Ayad, N. Bourouba, M. Benziane, I. Griche, S.S.M. Ghoneim, E. Ali, M. Elsisi, R.N.R. Ghaly, An efficient method for faults diagnosis in analog circuits based on machine learning classifiers, Alex. Eng. J. 77 (2023) 109–125, https://doi.org/10.1016/j.aej.2023.06.090.

[18] A. Silva Santos, R.J. da Silva, P.A. Montenegro, L.T. Faria, M.L.M. Lopes, C. R. Minussi, Integrating autoencoders to improve fault classification with PV system insertion, Elec. Power Syst. Res. 242 (2025) 111426, https://doi.org/10.1016/j.epsr.2025.111426.

[19] X. Li, Y. Xie, Analog circuits fault detection using cross-entropy approach, J. Electron. Test. 29 (2013) 115–120, https://doi.org/10.1007/s10836-012-5344-x.

[20] Y. Li, R. Zhang, Y. Guo, P. Huan, M. Zhang, Nonlinear soft fault diagnosis of analog circuits based on RCCA-SVM, IEEE Access 8 (2020) 60951–60963, https://doi.org/10.1109/ACCESS.2020.2982246.

[21] L.A.E. Noussaiba, F. Abdelaziz, ANN-based fault diagnosis of induction motor under stator inter-turn short-circuits and unbalanced supply voltage, ISA Trans. 145 (2024) 373–386, https://doi.org/10.1016/j.isatra.2023.11.020.

[22] A.R. Nasser, A.T. Azar, A.J. Humaidi, A.K. Al-Mhdawi, I.K. Ibraheem, Intelligent fault detection and identification approach for analog electronic circuits based on fuzzy logic classifier, Electronics 10 (2021) 2888, https://doi.org/10.3390/electronics10232888.

[23] A. Arabi, N. Bourouba, A. Belaout, M. Ayad, An accurate classifier based on adaptive neuro-fuzzy and features selection techniques for fault classification in analog circuits, Integration 64 (2019) 50–59, https://doi.org/10.1016/j.vlsi.2018.08.001.

[24] M. Tadeusiewicz, S. Hałgas, A method for multiple soft fault diagnosis of linear analog circuits, Meas. J. Int. Meas. Confed. 131 (2019) 714–722, https://doi.org/10.1016/j.measurement.2018.09.001.

[25] A. Moezi, S.M. Kargar, Simultaneous fault localization and detection of analog circuits using deep learning approach, Comput. Electr. Eng. 92 (2021), https://doi.org/10.1016/j.compeleceng.2021.107162.
[26] D. Binu, B.S. Kariyappa, A survey on fault diagnosis of analog circuits: taxonomy and state of the art, AEU - Int. J. Electron. Commun. 73 (2017) 68–83, https://doi.org/10.1016/j.aeue.2017.01.002.
[27] V. Khemani, M.H. Azarian, M. Pecht, WavePHMNet: a comprehensive diagnosis and prognosis approach for analog circuits, Adv. Eng. Inform. 59 (2024) 102323, https://doi.org/10.1016/j.aei.2023.102323.
[28] M. Sheikhan, A.A. Sha'bani, PSO-optimized modular neural network trained by OWO-HWO algorithm for fault location in analog circuits, Neural Comput. Appl. 23 (2013) 519–530, https://doi.org/10.1007/s00521-012-0947-9.
[29] S. Zhao, X. Liang, L. Wang, H. Zhang, G. Li, J. Chen, A fault diagnosis method for analog circuits based on EEMD-PSO-SVM, Heliyon 10 (2024) e38064, https://doi.org/10.1016/j.heliyon.2024.e38064.
[30] M.I. Dieste-Velasco, Application of a pattern-recognition neural network for detecting analog electronic circuit faults, Mathematics 9 (2021) 3247, https://doi.org/10.3390/math9243247.
[31] T. Zhong, J. Qu, X. Fang, H. Li, Z. Wang, The intermittent fault diagnosis of analog circuits based on EEMD-DBN, Neurocomputing 436 (2021) 74–91, https://doi.org/10.1016/j.neucom.2021.01.001.
[32] J. Yang, T. Gao, S. Jiang, A dual-input fault diagnosis model based on SE-MSCNN for analog circuits, Appl. Intell. 53 (2023) 7154–7168, https://doi.org/10.1007/s10489-022-03665-3.
[33] Y. Miao, Y. Zhang, F. Chen, Z. Wang, Analog circuit incipient fault detection based on attention mechanism and fully convolutional network, IEEE Access 12 (2024), https://doi.org/10.1109/ACCESS.2024.3403908.
[34] L. Shi, Y. Zhu, Y. Zhang, Z. Su, Fault diagnosis of signal equipment on the Lanzhou-Xinjiang high-speed railway using machine learning for natural Language processing, Complexity 2021 (2021), https://doi.org/10.1155/2021/9126745.
[35] A. Fazli, J. Poshtan, Wind turbine fault detection and isolation robust against data imbalance using KNN, Energy Sci. Eng. 12 (2024) 1174–1186, https://doi.org/10.1002/ese3.1706.
[36] A. Chahal, P. Gulia, N.S. Gill, J.M. Chatterjee, Performance analysis of an optimized ANN model to predict the stability of smart grid, Complexity (2022) 2022, https://doi.org/10.1155/2022/7319010.
[37] A. Afia, F. Gougam, W. Touzout, C. Rahmoune, H. Ouelmokhtar, D. Benazzouz, Spectral proper orthogonal decomposition and machine learning algorithms for bearing fault diagnosis, J. Brazilian Soc. Mech. Sci. Eng. 45 (2023) 550, https://doi.org/10.1007/s40430-023-04451-z.
[38] A. Kumar, A.P. Singh, Fuzzy classifier for fault diagnosis in analog electronic circuits, ISA Trans. 52 (2013) 816–824, https://doi.org/10.1016/j.isatra.2013.06.006.
[39] W. Yu, Y. Sui, J. Wang, The faults diagnostic analysis for analog circuit based on FA-TM-ELM, J. Electron. Test. 32 (2016) 459–465, https://doi.org/10.1007/s10836-016-5597-x.
[40] M. Parai, S. Srimani, K. Ghosh, H. Rahaman, Multi-source data fusion technique for parametric fault diagnosis in analog circuits, Integration 84 (2022) 92–101, https://doi.org/10.1016/j.vlsi.2022.01.005.
[41] P. Bilski, Hierarchical diagnostics of analog systems based on the ambiguity groups detection, Measurement 119 (2018) 1–10, https://doi.org/10.1016/j.measurement.2018.01.029.
[42] G. Zhao, X. Liu, B. Zhang, Y. Liu, G. Niu, C. Hu, A novel approach for analog circuit fault diagnosis based on Deep Belief network, Measurement 121 (2018) 170–178, https://doi.org/10.1016/j.measurement.2018.02.044.
[43] C. Zhang, Y. He, T. Yang, B. Zhang, J. Wu, An analog circuit fault diagnosis approach based on improved wavelet transform and MKELM, circuits, syst, Signal Process. 41 (2022) 1255–1286, https://doi.org/10.1007/s00034-021-01842-2.
[44] The MathWorks Inc, Statistics and Machine Learning Toolbox™ User's Guide ©Copyright 2004–2022, The MathWorks, Inc., 2022.
[45] The MathWorks Inc., Deep Learning Toolbox™ User's Guide (R2022B), 2022.
[46] A.J. Larner, The 2x2 Matrix, Springer International Publishing, Cham, 2021, https://doi.org/10.1007/978-3-030-74920-0.
[47] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer New York, New York, NY, 2009, https://doi.org/10.1007/b94608.
[48] G. Dougherty, Pattern Recognition and Classification, Springer New York, New York, NY, 2013, https://doi.org/10.1007/978-1-4614-5323-9.
[49] B.W. Matthews, Comparison of the predicted and observed secondary structure of T4 phage lysozyme, Biochim. Biophys. Acta Protein Struct. 405 (1975) 442–451, https://doi.org/10.1016/0005-2795(75)90109-9.
[50] S. Szabó, I.J. Holb, V.É. Abriha-Molnár, G. Szatmári, S.K. Singh, D. Abriha, Classification assessment tool: a program to measure the uncertainty of classification models in terms of class-level metrics, Appl. Soft Comput. 155 (2024), https://doi.org/10.1016/j.asoc.2024.111468.
[51] D. Chicco, G. Jurman, A statistical comparison between matthews correlation coefficient (MCC), prevalence threshold, and fowlkes–Mallows index, J. Biomed. Inf. 144 (2023) 104426, https://doi.org/10.1016/j.jbi.2023.104426.
[52] B. Thiyam, S. Dey, Efficient Feature Evaluation approach for a class-imbalanced dataset using machine learning, Procedia Comput. Sci. 218 (2022) 2520–2532, https://doi.org/10.1016/j.procs.2023.01.226.