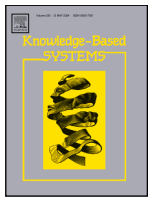
Knowledge-Based Systems 328 (2025) 114203

Contents lists available at ScienceDirect

Knowledge-Based Systems

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



An effective multi-objective metaheuristic for the support vector machine with feature selection

Mathias Badilla-Salamanca, Rosa Medina Durán, Carlos Contreras-Bolton *

Departamento de Ingeniería Industrial, Universidad de Concepción, Edmundo Larenas 219, Concepción, 4070409, Chile

ARTICLE INFO

Keywords:
Support vector machine
Feature selection
Multi-objective optimization
NSGA-II
Parameter tuning

ABSTRACT

Feature selection (FS) is crucial in supervised learning, mainly when dealing with high-dimensional datasets, since the models' efficiency decreases due to the curse of dimensionality. Reducing the number of features enhances computational efficiency and improves models' interpretability and generalization. Support vector machine (SVM) has been widely used with FS due to its ability to assign importance to individual features through feature-specific regressors, which can be eliminated when deemed irrelevant. This paper proposes a multi-objective metaheuristic approach based on non-dominated sorting genetic algorithm II, integrating FS into the soft-margin SVM model to optimize both predictive performance and computational efficiency. Unlike prior methods with static FS, our approach dynamically selects features to approximate the Pareto-optimal frontier, balancing structural and empirical risk. The proposed algorithm incorporates a novel solution representation, specialized crossover and mutation operators, and a weighted optimization strategy to effectively handle dataset imbalances. Additionally, we apply effective parameter tuning based on three considered performance metrics, resulting in three distinct versions of our approach, each exhibiting different search behaviors. Extensive experiments on well-known binary classification datasets demonstrate that all three versions outperform the state-of-the-art algorithm in both predictive performance and computational efficiency within the given time limits. Among them, the version that employs a conservative FS strategy, maintaining larger feature subsets while applying high mutation rates, achieved the best overall results. In addition, our approach also exhibits competitive performance on real-world large-scale datasets.

1. Introduction

Supervised classification has become increasingly prevalent in the analysis of real-world datasets. Applying specific assumptions, mainly statistical, enables the creation of models that help interpret data and anticipate future outcomes. This approach facilitates the identification of patterns and trends within existing data, supporting informed decision-making and predicting outcomes in previously unobserved scenarios.

In supervised learning, several algorithms are commonly employed to perform classification tasks. Among them is the support vector machine (SVM) algorithm [1], which aims to identify an optimal hyperplane that separates the feature space, effectively distinguishing between classes. SVMs play a crucial role in various fields, from medicine to computer vision, contributing significantly to solving real-world problems [2].

SVMs are often trained on large datasets, where not all features contribute valuable information to the model, and their efficiency decreases due to the curse of dimensionality. This well-known phenomenon arises when the number of features grows significantly relative to the number of observations. As dimensionality increases, data points become sparser in the feature space, making it more challenging for machine learning algorithms, such as SVMs, to generalize patterns and effectively distinguish between classes [3,4]. This sparsity reduces model effectiveness, increases computational complexity, and heightens the risk of overfitting, as the algorithm may capture noise rather than meaningful patterns.

The curse of dimensionality creates the need for algorithms capable of fast training and algorithms that can select features without compromising prediction quality. Feature selection (FS) is a widely used technique to mitigate the curse of dimensionality in supervised learning. FS identifies the most relevant features while eliminating redundant or irrelevant ones. By reducing the number of dimensions, FS improves model efficiency, enhances interpretability, and prevents performance degradation due to excessive feature complexity [5–7].

Traditional research has primarily focused on achieving optimal classification performance for a given dataset, often relying on single-objective optimization techniques. However, recent studies have explored multi-objective optimization for SVMs, aiming to generate a

* Corresponding author.
E-mail addresses: mbadilla2018@udec.cl (M. Badilla-Salamanca), rosmedina@udec.cl (R. Medina Durán), carlos.contreras.b@udec.cl (C. Contreras-Bolton).

https://doi.org/10.1016/j.knosys.2025.114203
Received 26 February 2025; Received in revised form 13 July 2025; Accepted 29 July 2025
Available online 5 August 2025

diverse set of Pareto-optimal solutions. This frontier provides the decision maker with a rich set of alternative solutions to make a better decision. Recent works, such as Alcaraz et al. [8], Valero-Carreras et al. [9], and Alcaraz [10], have successfully integrated multi-objective metaheuristics, particularly the non-dominated sorting genetic algorithm II (NSGA-II), into SVM with FS (SVM-FS). These studies have demonstrated that leveraging multi-objective optimization strategies can significantly improve predictive accuracy and computational efficiency.

These improvements are particularly significant due to the opportunity costs associated with long computing times, delays in obtaining results, and the financial burden of computing resources (e.g., server rental, electricity). Furthermore, enhancing predictive accuracy facilitates more informed decision-making in critical fields such as finance, banking, logistics, and high-stakes sectors like healthcare. While existing NSGA-II approaches for SVM-FS offer a starting point, they suffer certain significant drawbacks. Firstly, their solution representation inherently constrains the search space, preventing the exploration of numerous potentially optimal feature subsets. Secondly, they often impose a rigid constraint on the number of selected features, limiting their adaptability to datasets with varying feature importance. Finally, the high computational cost associated with these methods renders them impractical for tackling many real-world instances.

To overcome these limitations, this paper introduces an effective multi-objective metaheuristic that seamlessly integrates SVM and FS. Our approach prioritizes both maximizing predictive performance and significantly enhancing computational efficiency. Specifically, we propose a novel solution representation that directly encodes hyperplanes, thereby expanding the search space and enabling the discovery of more diverse solutions. Furthermore, we incorporate customized genetic operators designed specifically for this representation, substantially reducing the computing time of our NSGA-II implementation. This allows our method to handle large-scale problems efficiently. Extensive experiments on well-known binary classification datasets demonstrate that our approach outperforms state-of-the-art algorithm in both predictive performance and computational efficiency within the given time limits. Additionally, we apply an effective parameter tuning based on three considered performance metrics to improve our algorithm's performance further, resulting in three distinct versions of our approach, each exhibiting different search behaviors. Moreover, our method also shows competitive performance on real-world large-scale datasets, highlighting its scalability and practical applicability across diverse problem settings.

The paper's structure is as follows. Section 2 provides a literature review of methods that address SVM-FS and presents the main contributions of our study. The problem and its related mathematical models are formally defined in Section 3. The proposed multi-objective metaheuristic, along with its components and detailed explanation, is described in Section 4. Section 5 presents and discusses the extensive computational experiments. Finally, the conclusions and directions for future work are summarized in Section 6.

2. Literature review

The literature review focuses on studies related to FS in SVM, with particular emphasis on models and algorithms based on the soft-margin SVM model. This SVM variant allows classification error tolerance, enabling it to handle data that are not perfectly separable [1,11].

Studies utilizing the soft-margin SVM model for FS have predominantly employed mathematical programming models and, recently, multi-objective metaheuristics. A seminal contribution in this area was made by Maldonado et al. [12], who pioneered addressing the challenge of combining SVM-FS through an exact approach based on the soft-margin SVM model. They proposed FS as an NP-hard problem and introduced two mixed integer linear programming (MILP) formulations to optimize FS and classifier construction simultaneously. Their approach demonstrated that reducing the number of relevant features can significantly enhance classifier performance compared to traditional methods.

Another of the exact approaches for SVM-FS was the method based on generalized Benders decomposition introduced by Aytug [13]. This approach allowed the decomposition of the original problem into a master problem and tractable subproblem. The master problem determines the subset of features for the SVM, while the subproblem solves the SVM training problem using only the selected features. This approach showed considerable sensitivity to the calibration of the features penalty parameter.

Later, Gaudioso et al. [14] explored Lagrangian relaxation as an alternative approach to enhance the efficiency of SVM-FS. This technique introduces greater flexibility by decomposing the problem into a master problem and a subproblem. The master problem identifies the optimal subset of features, while the subproblem trains the SVM model using only the selected features. This structure enables the incorporation of constraints that penalize the selection of excessive features, promoting a more compact model. The method stands out for its ability to balance model complexity and predictive performance. However, its effectiveness critically relies on the appropriate parameter tuning.

Benítez-Peña et al. [15] proposed a cost-sensitive approach to FS in SVM, addressing scenarios where acquiring certain features is difficult or expensive, balancing accuracy and FS costs. The approach followed a two-step process: first, a MILP model selects features based on desired true positive and true negative rates; second, a mixed integer convex quadratic programming (MICQP) model with cost-sensitive constraints fits SVMs using either linear or radial kernels. The main idea lies in shifting from traditional margin maximization to minimizing the number of selected features while enforcing upper bounds on false positive and negative rates.

In parallel, Labbé et al. [16] improved the work of Maldonado et al. [12] by proposing a MILP formulation that incorporates a budget constraint and tightens the bounds on the separating hyperplane coefficients, limiting the number of features used in the classification process. Their study emphasizes the formulation's effectiveness, presenting exact and heuristic methods to solve the proposed problem. These methods were validated through comparisons with traditional classification techniques, demonstrating strong performance and efficiency.

Lee et al. [17] extended the SVM-FS framework to account for feature costs by incorporating a budget constraint, aiming to maintain classification accuracy while prioritizing using less expensive features. To achieve this, they formulated MILP models and a robust counterpart to address uncertainties in feature costs. Experimental studies on various benchmark and synthetic datasets demonstrate that their proposed MILP models achieved competitive results in terms of predictive and economic performance.

Baldomero-Naranjo et al. [18] presented an SVM-based model that simultaneously tackles outlier detection and FS. The model incorporates a rampant loss error margin and a budget constraint to limit the number of selected features. Its formulation is modeled as a MILP model using big-M parameters. The authors proposed two approaches to solve this formulation: an exact and heuristic method. The effectiveness of the heuristic approach was validated by comparing its solution quality with that of the exact method. Additionally, the classifiers obtained through the heuristic approach were tested on real datasets and benchmarked against existing SVM-based models, demonstrating their efficiency and practical applicability.

Recent research has complemented the development of MILP models for FS. Studies by Alcaraz et al. [8], Valero-Carreras et al. [9], and Alcaraz [10] introduce an NSGA-II to address the SVM-FS from a multi-objective perspective. This approach enables the specification of the desired number of features and automatically identifies which ones to include, simultaneously optimizing both the SVM model's performance and FS. These studies mark a significant evolution in FS techniques, leveraging multi-objective strategies that integrate classifier optimization with efficient FS, resulting in more balanced and effective solutions.

First, Alcaraz et al. [8] compared NSGA-II with the augmented ϵ-constraint 2 (AUGMECON2) method, an exact lexicographic optimization algorithm introduced by Mavrotas and Florios [19]. AUGMECON2 follows a hierarchical approach, seeking optimal solutions for a primary objective and only considering secondary objectives when multiple equally optimal solutions exist for the primary one. The comparison focuses on evaluating computational efficiency and the algorithms' ability to identify superior solutions within the specific context of SVM-FS.

Then, Valero-Carreras et al. [9] modified the first objective in the work of Alcaraz et al. [8], replacing it with the number of misclassified vectors. Their study evaluated NSGA-II in terms of predictive quality in FS. This extension broadens the scope of the approach, shifting the focus beyond simply dominating solutions to enhancing predictive accuracy. It establishes a connection between combinatorial optimization techniques and artificial intelligence research, highlighting the synergy between these fields.

Finally, Alcaraz [10] built on the same paradigm as its predecessor [8] but introduces a complete redesign. The study proposed a new encoding scheme and efficient procedures tailored to address the primary limitations of the earlier approach. These improvements aim to reduce the search space, exclude non-promising solutions, and enhance the algorithm's ability to solve instances for a broader range of parameter values. Additionally, the redesign mitigates the high computational effort required to evaluate solutions, which had previously slowed down the technique. Computational experiments demonstrated that the new method outperforms the results achieved by its NSGA-II predecessor, offering better performance and efficiency.

Although this revision mainly focuses on models and algorithms based on the soft-margin SVM model, it is important to highlight an alternative research avenue within SVM-FS. This alternative research simultaneously optimizes SVM model parameters and FS to enhance predictive performance and computational efficiency. For instance, Bouraoui et al. [20] employed a multi-objective genetic algorithm to fine-tune SVM parameters while selecting an optimal subset of features. Similarly, Faris et al. [21] used a metaheuristic approach to optimize both FS and SVM parameters, achieving high accuracy while significantly reducing the number of features. In applied contexts, Candelieri et al. [22] used parallel global optimization to tune SVM hyperparameters, improving computational efficiency. Sabzekar and Aydin [23] propose a noise-aware FS method that combines relaxed SVM with a sequential backward search to improve classification accuracy under noisy conditions. Additionally, Dudzik et al. [24] proposed an evolutionary technique to optimize SVM models for large, imbalanced datasets, enhancing performance under challenging conditions. Hybrid approaches as Huang et al. [25], Abasabadi et al. [26], Jain and Jain [27], and recent other approaches continue to advance this field, as seen in these works [28–31].

In addition to the mathematical programming models and metaheuristic-based approaches reviewed above, recent advancements have explored deep learning-based methods for FS. These methods have shown promising results across various domains, including bioinformatics, computer vision, and natural language processing. These methods leverage neural networks' powerful representation learning capabilities to identify the most informative features from high-dimensional data. Unlike the SVM-FS models discussed previously, deep learning-based FS methods can capture complex, non-linear relationships between features and target variables. FS in deep learning is typically integrated into the model training process using techniques such as attention mechanisms [32,33], sparsity-inducing regularizations [34], autoencoders [35], or transfer learning combined with convolutional neural networks [36], or reinforcement learning [37]. These methods aim to reduce redundancy, improve generalization, and enhance interpretability without sacrificing predictive performance. However, despite their flexibility, these methods often have higher computational costs, potential interpretability challenges, and a greater risk of overfitting in small-sample settings.

Table 1
A summary of the referenced works focusing specifically on soft-margin SVM-FS.

Article	Method	Number of features	Number of objectives
Maldonado et al. [12]	MILP	Fixed	1
Aytug [13]	MILP	Fixed and dynamic	1
Gaudioso et al. [14]	MILP	Dynamic	1
Benítez-Peña et al. [15]	MILP and MICQP	Dynamic	1
Labbé et al. [16]	MILP and heuristic	Fixed	1
Lee et al. [17]	MILP	Dynamic	1
Baldomero-Naranjo et al. [18]	MILP and heuristic	Dynamic	1
Alcaraz et al. [8]	AUGMECON2 and NSGA-II	Fixed (5)	2
Valero-Carreras et al. [9]	NSGA-II	Fixed (5)	2
Alcaraz [10]	NSGA-II	Fixed (5)	2
This work	NSGA-II	Dynamic	2

A summary of the referenced works focusing specifically on soft-margin SVM-FS is provided in Table 1, detailing the problems addressed, the methods employed, the static or dynamic treatment of the number of features, and the number of objectives considered. The table shows the diversity of exact methods used to tackle the problem and how these approaches integrate different perspectives on FS, treating the number of features as fixed parameters or dynamic.

The literature review indicates that SVM-FS has mainly been addressed using exact methods and few metaheuristic approaches. Building upon the metaheuristic approach of Alcaraz [10], we propose an effective multi-objective algorithm based on NSGA-II to approximate the Pareto-optimal frontier for structural and empirical errors while simultaneously selecting features. Our approach leverages the soft-margin SVM model with integrated FS. The main contributions of this study are as follows:

- Our algorithm employs a direct representation of hyperplanes to partition the solution space, significantly reducing computing time per solution. In contrast to Valero-Carreras et al. [9], which selects vectors from the dataset and computes intersecting hyperplanes following the method in Alcaraz et al. [8], our approach avoids this additional computation. Furthermore, while Alcaraz [10] expands their encoding to cover a broader search space, our method ensures the representation of every possible solution.
- We introduce a crossover operator and two mutation operators tailored to our representation. In Alcaraz et al. [8], the authors propose separate crossover and mutation operators for features and vectors. Similarly, Alcaraz [10] separate both vectors (class) and features into crossover operations, while their mutation operator modifies vectors, hyperplane coordinates, and features independently. In contrast, we adopt an adapted partially mapped crossover for FS and use the intercept's average. Our mutation operators refine the feature subset by adjusting feature weights and intercepts or modifying the number of selected features.
- To mitigate dataset imbalances, we incorporate a weighted optimization strategy. Unlike Alcaraz et al. [8] and Alcaraz [10], our approach includes a weight for classification errors based on class prevalence in the second objective.
- Our approach dynamically adjusts the number of selected features to converge toward an optimal subset. In contrast to Valero-Carreras et al. [9] and Alcaraz [10], which rely on a fixed FS, our method allows for adaptive refinement throughout the optimization process.
- We optimize our algorithm through parameter tuning based on three performance metrics, leading to three distinct versions, each exhibiting unique search behaviors.
- Extensive experiments were conducted on both well-known binary classification datasets and real-world large-scale datasets. Results

demonstrate that all three versions outperform the state-of-the-art algorithm in both predictive accuracy and computational efficiency within given time constraints. Notably, the version employing a conservative FS strategy, retaining larger feature subsets while applying high mutation rates, achieved the best overall performance.

3. Support vector machine with feature selection

The FS problem focuses on identifying a subset of relevant features from a dataset by eliminating redundant, noisy, or irrelevant features. FS is essential because including more features does not always lead to better analytical results and can even degrade model performance. Additionally, an excessive number of features increases the computational cost, making data processing more time-consuming.

SVM is an artificial intelligence model introduced by Cortes and Vapnik [1] and Vapnik [38], designed for binary classification tasks by optimally separating an n-dimensional space with two hyperplanes. This approach builds on the theoretical foundations laid by Vapnik and Chervonenkis [39], who proposed a method for constructing an optimal separating hyperplane for pattern recognition. Initially, applying it to separable cases in Vapnik and Chervonenkis [40], where a linear hyperplane perfectly distinguished between two classes. Later, Vapnik [38] extended the concept to non-separable data by introducing the soft-margin SVM, which incorporates slack variables to handle classification errors effectively. This separation is achieved by defining a set of support vectors that determine a hyperplane capable of dividing high-dimensional input space, classifying data points as "above" or "below" this hyperplane. SVMs aim to optimize two key objectives: structural error (maximizing the margin) and empirical error (minimizing classification error).

The structural error relates to the distance between the two separating hyperplanes: the larger the distance (or margin), the lower the model's uncertainty, improving its ability to make accurate predictions on unseen data. Meanwhile, empirical error refers to the difference between the model's predicted and actual labels in the training data. Fig. 1 illustrates the concept of support vectors using a two-dimensional dataset, where vectors *i, j*, and *k* are needed to support the hyperplanes that divide the space. Additionally, the figure shows the margin between the hyperplanes, which is considered in the objective function, and the classification error of vectors *m* and *l*.

With the development of supervised learning algorithms such as SVM, the challenge of efficiently handling increasingly large datasets has arisen. Thus, the training and evaluating stage becomes computationally expensive as data increases in observations and features. To address this, several methodological advancements have been proposed. One such approach is the relaxed constraints SVM, which enhances the robustness and adaptability of the traditional SVM by incorporating mechanisms to handle noisy and uncertain data more effectively [41]. Another approach is the FS, which aims to address the problem of high dimensionality of datasets by reducing the number of input features while retaining the essential information to solve the issue. This not only improves computational efficiency but can also lead to more straightforward and generalizable models.

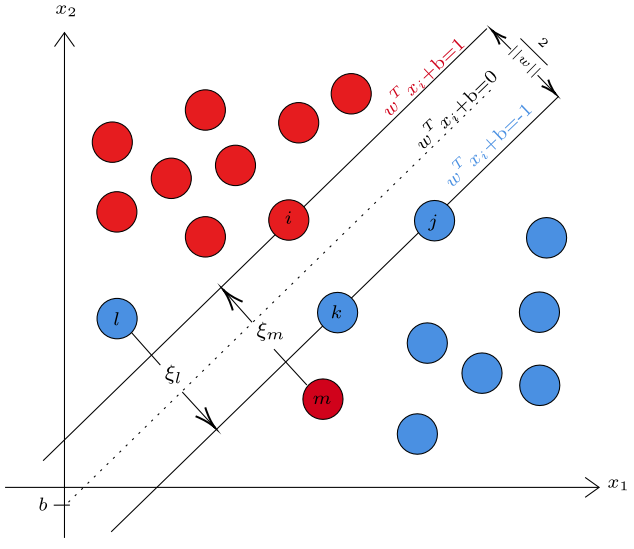


Fig. 1. SVM structure.

FS approaches are generally categorized into filter, wrapper, and embedded methods [42,43]. Filter methods evaluate each feature's relevance independently of the model. Wrapper methods use a predictive model to assess the quality of various feature subsets. Embedded methods integrate FS into the model training process, selecting relevant features for model construction. The FS approach in this research is an embedded method, as FS is performed simultaneously with model training due to the solution representation used in our proposed algorithm.

3.1. Mathematical programming model

The soft-margin SVM can be formulated as a convex quadratic programming model [1]. Formally, let $N = \{1, \dots, n\}$ be a set of vectors divided into two classes, $\{-1, 1\}$. Each vector $i \in N$ is represented by a pair $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, where d denotes the number of features analyzed for each vector in N, and the set $D = \{1, \dots, d\}$ represents these features. While x_i contains the features and y_i indicates the class membership (1 or -1). The decision variables include $w_j, j \in D$, representing the weight vector, b is the bias term, and ξ_i, $i \in N$, is a set of slack variables, representing the classification error. The objective of the SVM is to determine an optimal hyperplane $h(x) = w^T x + b$ that separates the two classes while minimizing classification error. The soft-margin SVM formulation proposed by Cortes and Vapnik [1] is presented below:

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{i \in N}\xi_i \tag{1}$$

subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i \in N \tag{2}$$

$$\xi_i \in \mathbb{R}_0^+ \quad \forall i \in N \tag{3}$$

$$w_j \in \mathbb{R} \quad \forall j \in D \tag{4}$$

$$b \in \mathbb{R} \tag{5}$$

The first term of the objective function (1) corresponds to the structural risk of the model since $\frac{2}{\|w\|}$ represents the distance between the hyperplanes. The second term refers to the empirical risk and seeks to minimize the misclassified vectors multiplied by a hyperparameter C. Thus, C grants a weighting of importance between both objectives. Constraints (2) correspond to the definition of both hyperplanes. Therefore, the vector y indicates whether the training vector is of the positive or negative class, and when multiplied by the evaluation of x, it indicates whether it is well or poorly classified. In a misclassification, the error is added to the decision variable ξ, penalizing the objective function. Finally, constraints (3)–(5) correspond to the domain of the decision variables.

Building on the previous model, the SVM-FS can be formulated as an MICQP model proposed by Alcaraz et al. [8]. This formulation introduces a binary decision variable t_j $(j \in D)$ to select the features included in the solution and a fixed parameter $p < d$ to limit the number of features considered. Therefore, the MICQP model extends the previous formulation with the following additions:

$$\sum_{j \in D} t_j = p \tag{6}$$

$$\left|w_j\right| \leq M t_j \quad \forall j \in D \tag{7}$$

$$t_j \in \{0, 1\} \quad \forall j \in D \tag{8}$$

Constraint (6) ensures that exactly p features are selected from the total available features d. The binary decision variable t_j determines whether a feature j is included in the model ($t_j = 1$) or excluded ($t_j = 0$). The sum of all selected features must equal the fixed parameter p, effectively limiting the number of active features in the solution. Constraints (7) link the selection of a feature j (t_j) with its corresponding weight variable w_j. If a feature is not selected ($t_j = 0$), the constraint forces its weight to be zero ($w_j = 0$). If a feature is selected ($t_j = 1$), its weight w_j is allowed to take any value within a valid range determined by a sufficiently large constant M. This mechanism ensures that only selected features contribute to the decision boundary of the SVM model. Constraints (8) are the domain of the decision variables t_j.

Extending the MICQP model for SVM-FS, Alcaraz et al. [8] propose a multi-objective approach by redefining the objective function (1) into two separate objectives, (9) and (10), effectively removing the need for the parameter C.

$$\text{minimize } f_1 = \frac{1}{2}\|w\|^2 \tag{9}$$

$$\text{minimize } f_2 = \sum_{i \in N} \xi_i \tag{10}$$

This paper builds on the multi-objective MICQP model proposed by Alcaraz et al. [8], considering modifying the objective f_2 in (10) by replacing it with the objective presented in (11), that contemplates the addition of the parameter α_i ($i \in N$), which weights classification errors according to the class of each misclassified vector based on its prevalence suggested by Yang et al. [44]. For example, in an instance where 70 % of the vectors belong to class 1 and 30 % to class -1, the parameter would assign a weight of 0.7 to errors in class 1 and a weight of 0.3 to errors in class -1.

Furthermore, the original constraint (6), which relies on a fixed parameter p, is replaced with a dynamic constraint that maintains the same left-hand side but introduces an upper bound based on the total number of features (d), as specified in (12). With these modifications, we can dynamically adjust the number of features considered and try to converge to an ideal feature subset. This contrasts with Valero-Carreras et al. [9], which uses p as a number fixed of five features.

$$\text{minimize} \quad f_2 = \sum_{i \in N} \alpha_i \xi_i \tag{11}$$

$$\sum_{j \in D} t_j \leq d \tag{12}$$

Therefore, the model considered in this work considers the constraints (2)–(5), (7), (8), (12), and the objective functions (9) and (11). Thus, this model allows selecting between one or all of the dataset's features. Consequently, it is possible to notice that the computational complexity of this problem increases since the ways of selecting the features grow exponentially according to $\sum_{i=1}^{d} \binom{d}{i}$, and simultaneously, the model performs the assignment of the weights for each selected feature. Additionally, this formulation presents challenges in determining when and how many features to remove or add, as excessive feature elimination may lead to overfitting and degrade model generalization.

Given this problem's exponential complexity and multi-objective nature, which requires optimizing multiple conflicting objectives simultaneously, exact methods are not viable due to their high computational cost and impracticality for large datasets. This justifies using metaheuristic approaches, which provide a scalable and efficient alternative by heuristically exploring the solution space instead of relying on exhaustive searches. Metaheuristics enable a balanced trade-off between solution quality and computational feasibility, making them well-suited for tackling the challenges of multi-objective optimization of the two objectives of the problem.

4. Proposed algorithm

NSGA-II is a well-known multi-objective evolutionary algorithm [45]. The algorithm generates an initial population of solutions and iteratively applies genetic operators such as selection, crossover, and mutation to evolve the population. A key feature of NSGA-II is its ability to maintain a diverse population of non-dominated solutions, ensuring good coverage of the Pareto front. This makes it particularly effective for solving multi-objective problems that involve two or more objectives.

The algorithm is built on the principle of non-dominance to generate and maintain a set of solutions for multiple objectives. Non-dominance between two solutions means that neither can be considered absolutely superior, as one may perform better in some objectives but worse in others. In contrast, dominance implies that one solution is superior in all objectives compared to another. A set of non-dominated solutions forms what is known as the Pareto frontier, where all solutions share the same indifference curve [46]. If no other solution dominates the solutions on this frontier, they represent the optimal trade-offs between conflicting objectives.

4.1. Overall algorithm

This section presents our proposed algorithm, which consists of an NSGA-II. We also describe the NSGA-II's evolutionary process, solution representation, initial population generation, selection, crossover, and mutation operators.

Algorithm 1: NSGA-II.

```
   Output: P
1  P ← initial-population (K)
2  evaluate-population (P)
3  F ← fast-non-dominated-sort (P)
4  t ← 0
5  while a given time limit T is not exceeded do
6      Q ← {∅}
7      for k ← 1 to K do
8          (a, e) ← selection-dominated-tournament (P)
9          i ← crossover (a, e)
10         if random(0, 1) ≤ ρ_t^1 then
11             i ← mutation-1(i)
12         if random(0, 1) ≤ ρ_t^2 then
13             i ← mutation-2(i)
14         Q ← Q ∪ {i}
15     evaluate-population (Q)
16     F ← fast-non-dominated-sort (P ∪ Q)
17     P ← {∅}
18     P ← ⋃_{k=1}^{m} F_k    where m = max{k : Σ_{j=1}^{k} |F_j| ≤ K}
19     if |P| < K then
20         P ← P ∪ selection-crowding-distance (F_{m+1})
21     t ← t + 1
```

The functioning of our NSGA-II is described in Algorithm 1. It begins with creating the initial population P, of $\mathcal{K}$ individuals, which are evaluated with the considered two objective functions (lines 1–2). Once the population has been evaluated, the different non-dominance frontiers must be determined. These frontiers are used later in the selection and crossover operators to define the individuals suitable to prevail during the later generations. For this, the operator `fast-non-dominated-sort` is applied until all individuals are classified into a frontier (line 3). This operator ranks solutions based on Pareto dominance, efficiently partitioning the population into different non-dominated fronts. Where solutions in the first front are not dominated by any others, and subsequent fronts are ranked according to the number of solutions dominating them. This sorting mechanism enables effective selection pressure towards the Pareto-optimal front while maintaining diversity in the population. The proposed algorithm then enters its main loop (line 5), which consists

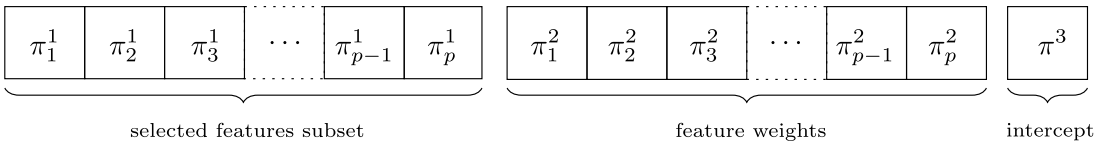


Fig. 2. Solution representation of the individuals.

of the evolutionary cycle. $\mathcal{K}$ new individuals are created (lines 7–14). To create each new individual, two individuals are randomly selected from the population and subjected to a dominance-based tournament to determine who will be parents. Then, the selected parents undergo crossover to produce an offspring, which may then go through to one, two, or no mutation operators (lines 9–13), depending on the mutation probabilities (ρ_t^1 and ρ_t^2, $t = \{0, 1, 2, \ldots\}$). After generating the new set of $\mathcal{K}$ individuals, this new population is evaluated (line 14). Next, the `fast-non-dominated-sort` operator is applied to former and new solutions, $P \cup Q$, organizing all $2\mathcal{K}$ individuals into non-dominated fronts stored in F, which will advance to the next generation (line 16). Thus, the new population is initialized as an empty set with a maximum capacity of $\mathcal{K}$ individuals (line 17). To fill P, a loop sequentially adds entire non-dominated fronts F_k as long as adding them does not exceed the population size $\mathcal{K}$ (line 18). If additional individuals are needed after adding all fronts, individuals from the current front F_{m+1} are selected based on crowding distance, with those having the greatest distances chosen to prioritize diversity until the population reaches $\mathcal{K}$ individuals (lines 19–20). Finally, the generation counter t is incremented, and the process repeats for subsequent generations (line 21). Once the time limit is reached, the algorithm returns the final population containing individuals on the Pareto front.

4.2. Solution representation

Each individual in the population P is composed of three chromosomes (π^1, π^2, π^3). The first chromosome (π^1) contains the identifiers of the selected features in non-decreasing order, and its size varies within the interval $[4, d]$. The rationale behind this range is explained in the context of the crossover operator and the maximum number of features. The second chromosome (π^2) contains the weights for these selected features, $\pi^2 \in \mathbb{R}$, reflecting their influence on the final classification; then π^2 is matched in size to π^1. Finally, the third chromosome (π^3) indicates the model's intercept (b), being a single value. Fig. 2 shows an individual's solution representation with p features selected, where the feature vector contains integer identifiers corresponding to dataset features, while the weights vector and intercept are continuous values.

To illustrate the solution representation shown in Fig. 2, let $\pi^1 = (1, 2)$, $\pi^2 = (1, -1)$, and $\pi^3 = -1$. This solution representation indicates selecting features 1 and 2, with weights of 1 and -1, respectively, and an intercept value of -1. We can compute the hyperplane separating the solution space with these values of the three chromosomes as $1x_1 + (-1)x_2 - 1 = 0$. Therefore, this equation represents the dotted line in Fig. 1. The other two planes dividing the space are obtained by shifting the equation value from 0 to 1 and -1.

4.3. Initial population

In the first population, each of the three chromosomes of each individual is generated randomly as follows:

- π^1: The first chromosome randomly selects a subset of features from a predefined range $[f^l, f^u]$, ensuring that each feature has an equal probability of being chosen.
- π^2: The second chromosome assigns random values to the weights of the features selected in π^1, with each weight obtained from a predefined range $[w^l, w^u]$.
- π^3: The third chromosome randomly generates the intercept value within a predefined range $[b^l, b^u]$.

This initial population ensures diversity while adhering to the predefined constraints for features, weights, and intercept values.

4.4. Selection

We employ a tournament selection based on dominance as our selection operator, which prioritizes higher-quality individuals in terms of multi-objective optimization. This operator selects two solutions from the current population, a and e, to produce an offspring i, which is then added to the new population Q. A tournament is conducted between two randomly selected solutions from P to choose each parent. The tournament winner is determined by the following criteria, applied in sequence if ties occur in the preceding steps:

1. The winner is the individual who belongs to a lower-ranked Pareto frontier.
2. If both belong to the same frontier, the winner is the individual that dominates a greater number of other solutions.
3. If still tied, the winner is chosen randomly.

4.5. Crossover

Crossover is a genetic operator that combines information from two parent solutions to produce new offspring, enhancing search space exploration. This process enables offspring to inherit features from both parents, supporting convergence toward a set of non-dominated solutions in multi-objective optimization problems. We proposed a crossover operator based on an adaptation of the partially mapped crossover operator [47].

Our adapted crossover begins by dividing π^1 (containing features identifiers) and π^2 (containing weights) into l subdivisions, each with a length L, which allows for structured recombination that respects the sequence integrity of each parent's attributes, given that features identifiers repetition within the chromosome must be avoided for feasibility. Given two parent solutions, a and e, that generate a new offspring i. Then, the first L elements of π_a^1 are chosen and copied to chromosome π_i^1 of the offspring. Since the chromosome π^1 is a non-decreasing ordered vector, we then locate the first element in π_e^1 that is greater than the last feature identifier added from π_a^1. Starting from this point in π_e^1, we copy the following L elements to the offspring chromosome. If π_e^1 does not contain L unique feature identifiers to add, we copy as many

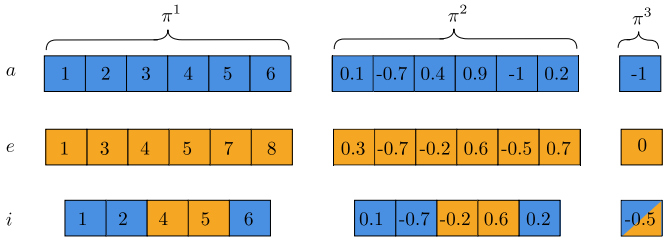


Fig. 3. Example of the proposed operator considering parent chromosomes a and e generate offspring i.

as possible and then continue filling the offspring with the remaining elements from π_a^1, ensuring that no feature identifiers are duplicated. The potential reduction in the size of π^1 in the offspring justifies using a minimum number of feature identifiers in the initial population.

Our crossover operator for the chromosome is briefly illustrated in Fig. 3, with a value of $L = 2$ for two parents, each of length 6. The selection process interleaves features from both parents. Specifically, the first third of the new chromosome is derived from parent *a*, the second third from parent *e*, and the final third provides to parent *a*. However, note that the last segment does not strictly adhere to $L = 2$, as the feature identifier 5 from parent *a* was already incorporated into the second third from parent *e*, leaving only element 6 to be added in the final segment. This complete process ensures that no elements are repeated, producing an ordered vector that combines feature identifiers from both parents. For the crossover of π^2, the process incorporates the selected features into the new individual by transferring the weight associated with each feature from the respective parent that contributed it. Finally, the crossover of chromosome π^3 is computed as the average of the intercepts of the two parents.

4.6. Mutation

Mutation enhances diversity within the population by randomly altering individuals' genes, enabling the exploration of new regions in the search space. This process helps prevent premature convergence by increasing the algorithm's ability to escape local optima and maintaining the population's genetic variety.

Our mutation process uses two operators, each with a different approach. The first operator modifies only the values of the individual's components, while the second ensures that the number of features comprising the individual remains unchanged. Both operators are described in detail below.

- *mutation-1*: This operator can replace a selected feature identifier with one not currently selected, it can modify the value of any weight by increasing or decreasing it by 5 %, and it can adjust the intercept value by increasing or decreasing it by 10 %. Note that every unselected feature has an equal probability of being chosen. Meanwhile, each of these changes has a 33 % probability of occurring and is not mutually exclusive. Consequently, while the expected number of mutations is one per operation, all three types of modifications can occur simultaneously.
- *mutation-2*: This operator introduces variability in the number of features an individual possesses, with an equal probability of increasing or decreasing the feature count. A discrete uniform distribution determines the specific number of features to add or remove. If features are to be added or removed, the change ranges from one feature up to a maximum of 20 % of the individual's current feature count. Feature removal is only applied if the individual has more than seven features, ensuring its solution's feasibility.

Both operators are applied in a non-exclusive manner, where the first operator has an initial probability of occurrence denoted by ρ_0^1 and the second by ρ_0^2. These probabilities gradually decrease over the iterations following the decay factor described in Eq. (13). The decay is controlled by the parameter B, set to 0.999, raised to the power of t, allowing for a progressive reduction in mutation probabilities. This results in increasingly localized exploration as the iterations (t) progress, drawing inspiration from the simulated annealing metaheuristic [48].

$$\rho_{t+1}^h = \rho_t^h \times B^t \qquad \forall h \in \{1,2\}, t \in \{0,1,2,\dots\} \tag{13}$$

4.7. Crowding distance

The crowding distance metric is used in NSGA-II to preserve diversity among solutions within the same Pareto front. This distance can be visualized as a measure of how isolated each solution is in a multidimensional objective space. Solutions with a high crowding distance are more spread out, while those with a low crowding distance are closer to other points, indicating denser areas in the solution space. Eq. (14) defines the crowding distance, where $d_j(F_k)$ represents the crowding distance of individual j within the frontier k, and $f_r(j)$ is the value of objective function r for individual j. Additionally, $\min_r(F_k)$ and $\max_r(F_k)$ denote the minimum and maximum values of objective function r for the frontier k, respectively.

$$d_j(F_k) = \frac{f_1(j+1) - f_1(j-1)}{\max_1(F_k) - \min_1(F_k)} + \frac{f_2(j+1) - f_2(j-1)}{\max_2(F_k) - \min_2(F_k)} \tag{14}$$

5. Computational experiments

The algorithm was implemented in Python 3.10. The experiments were conducted on the supercomputing infrastructure of the NLHPC, using a Lenovo ThinkSystem SR645 V3 node with two AMD EPYC 9754 processors with 2.25 GHz, each with 128 cores and running 768 GB RAM. All experiments were run with a single thread using the CentOS Linux 7 operating system (64-bit). We have made all benchmarking instances, along with the detailed results for each set of instances and the source code, available online at the following URL[1].

5.1. Dataset description

The proposed algorithm is evaluated through computational experiments on binary classification datasets of varying dimensions and feature sets commonly referenced in the literature. For comparison purposes, we benchmark our algorithm against the NSGA-II approach proposed by Valero-Carreras et al. [9], focusing exclusively on binary classification datasets utilized in their study. A brief description of each dataset is provided below.

- Arcene: This dataset was created for high-dimensional classification problems in bioinformatics. This dataset has gene expression profiles for cancer classification [49].
- Bioresponse: This dataset aims to predict the biological response of molecules based on their chemical properties, classifying responses as positive or negative [50].
- Duke: This dataset was designed for cancer detection and contains cell characteristics derived from biopsies, enabling the study of cancer-related patterns [51].
- German Credit: Containing data on credit applications in Germany, this dataset includes various characteristics of applicants. The objective is to predict whether a credit applicant is eligible based on their credit history and personal details [52].
- Gina Agnostic: This dataset focuses on handwritten digit recognition, where the task is to classify digits as even or odd. Only the unit digit provides meaningful information for this classification, while at least half of the features are distractors [53].
- Gisette: This dataset distinguishes between handwritten digits "4" and "9". It includes distractor features, making it helpful to evaluate FS methods [54].
- Ionosphere: This dataset is used to classify radar signals measured over the ionosphere. Each record represents a series of radar measurements reflected from the ionosphere, classifying the signals as "good" or "bad" to determine the relevance of objects detected on the radar [55].
- Wisconsin Breast Cancer (WBC): This dataset is designed to classify breast tumors as benign or malignant based on features extracted from digital biopsy images [56].

Arcene, German Credit, Gisette, Ionosphere, and, WBC datasets are available in the UCI Machine Learning Repository [57]. The Bioresponse

[1] https://github.com/maffijoule/MO-FS-SVM

Table 2
Characteristics of datasets.

Dataset	n	d
Arcene	900	10,000
Bioresponse	3000	1776
Duke	44	7129
German Credit	1000	20
Gina Agnostic	3468	970
Gisette	13,500	5000
Ionosphere	351	34
WBC	569	30

dataset can be accessed on the Kaggle platform [58], while the Duke dataset is available through LIBSVM [59]. Table 2 summarizes the main characteristics of each dataset, including their name, size (n), and features (d).

We limit our analysis to datasets specifically designed for binary classification, as some datasets in Valero-Carreras et al. [9] involve multiclass or regression tasks, which fall outside the primary focus of the SVM framework. While SVMs can be adapted for multiclass classification or regression, Valero-Carreras et al. [9] does not detail the strategies employed for such adaptations. Additionally, preprocessing details are not explicitly outlined in Valero-Carreras et al. [9]. To ensure a fair comparison, we analyze the entire dataset without relying on predefined training and testing splits, even when these are available. This decision mitigates potential inconsistencies between methodologies arising from the absence of clear preprocessing steps in the referenced study. By adhering to these considerations, we ensure our algorithm is comparable to the NSGA-II approach proposed by Valero-Carreras et al. [9].

The proposed algorithm applies data standardization for each instance by centering the features around their mean (μ) and scaling them according to their standard deviation (σ), as defined by Eq. (15). This ensures consistent normalization across all datasets.

$$z_i = \frac{x_i - \mu}{\sigma} \quad \forall i \in N \tag{15}$$

Fig. 4 illustrates the class distribution for each dataset, highlighting the presence of imbalanced instances. A notable example is the German Credit dataset, where class 1 accounts for 70 % of the dataset's vectors. This significant imbalance justifies the inclusion of weighting with α_i ($i \in N$) the classification errors in Eq. 11, ensuring a more balanced and fair evaluation of the algorithm's performance.

5.2. Metrics used

The evaluation of the proposed algorithm is based on the three performance metrics used in Valero-Carreras et al. [9], with the primary aim of ensuring methodological consistency and enabling a fair comparison with prior results. These metrics are derived from the confusion matrix analysis, a fundamental tool for assessing classification models. Fig. 5 illustrates the confusion matrix, which serves as the basis for calculating and interpreting each metric.

The confusion matrix provides a comprehensive overview of a model's performance by comparing predicted and actual values [11]. It displays the counts of true positives, true negatives, false positives, and false negatives, enabling a detailed evaluation of classification results. From this matrix, metrics such as the area under the receiver operating characteristic (AUC-ROC) curve [60], the F-Score [61,62], and Cohen's Kappa coefficient (CKC) [63] are calculated, offering insights into various aspects of the model's accuracy, balance, and reliability.

- AUC-ROC: It curve illustrates the trade-off between the true positive rate (TPR) (Eq. (16)) and the false positive rate (FPR) (Eq. (17)) across various decision thresholds. The AUC-ROC quantifies the likelihood that the model assigns a higher probability score to a positive instance than to a negative one. This metric leverages class probabilities, calculated using a sigmoid function to estimate likelihoods. The AUC-ROC ranges from 0 to 1, with values closer to 1 indicating superior discriminatory ability between classes, signifying a model's effectiveness in distinguishing positive from negative instances.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{16}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{17}$$

- F-Score: It is the harmonic mean of Precision (Eq. (18)) and Recall (Eq. (19)), offering a balanced measure that considers both the accuracy of positive predictions and the model's ability to identify all positive instances. It ranges from 0 to 1, with values closer to 1 indicating better performance. The F1-Score is particularly useful when there is an uneven class distribution or when false positives and false negatives carry different costs. The formula for the F1-Score is provided in Eq. (20).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{18}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{19}$$

$$\text{F-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{20}$$

- CKC: It is a statistical metric that quantifies the agreement between two categorical classifications while accounting for the agreement expected by chance. Its value ranges from -1 to 1, where -1 indicates total disagreement, 0 represents agreement equivalent to random chance, and 1 denotes perfect agreement. This coefficient is particularly useful for assessing inter-rater reliability or classification accuracy in imbalanced datasets. The formula for calculating CKC is provided in Eq. (23).

$$p_o = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{21}$$

$$p_c = \frac{(\text{TP} + \text{FP}) \times (\text{TP} + \text{FN}) + (\text{TN} + \text{FN}) \times (\text{TN} + \text{FP})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})^2} \tag{22}$$

$$\text{CKC} = \frac{p_o - p_c}{1 - p_c} \tag{23}$$

5.3. Experiment setup

The experimental design replicates the methodology outlined by Valero-Carreras et al. [9]. Each of the eight instances is executed with two distinct time limits, 1200 s and 3600 s. Furthermore, five different training and testing splits are created using the K-fold cross-validation technique [64]. K-fold cross-validation involves dividing the dataset into equal-sized subsets. Following Valero-Carreras et al. [9], five splits are used. Each iteration selects one subset as the test set, while the remaining subsets are used for training. This process is repeated five times, ensuring each subset is used once as the test set. This cross-validation procedure mitigates the risk of overfitting, especially in datasets with a small number of samples and high dimensionality, such as Duke. By rotating the test subset across folds, the evaluation becomes more robust and less sensitive to specific data partitions, providing a more reliable estimate of the algorithm's generalization performance. We have run the algorithm three independent times, with different seeds, for each of the splits, obtaining a set of non-dominated solutions per run and selecting the best individual in the minimum Pareto frontier per metric. Fig. 6 illustrates process above described. In each split, a different subset is designated as the test set (highlighted in orange), while the remaining subsets serve as the training set (highlighted in green). Consequently, the experimental setup of our algorithm involves eight instances, five K-fold splits, three random seeds, and two time limits, resulting in a total of 240 runs.

5.4. Parameter tuning

The configuration of the proposed algorithm requires a large number of parameters to be set beforehand. Our approach considers nine parameters: $\mathcal{K}$ is the population size; f^l and f^u are lower and upper percentage

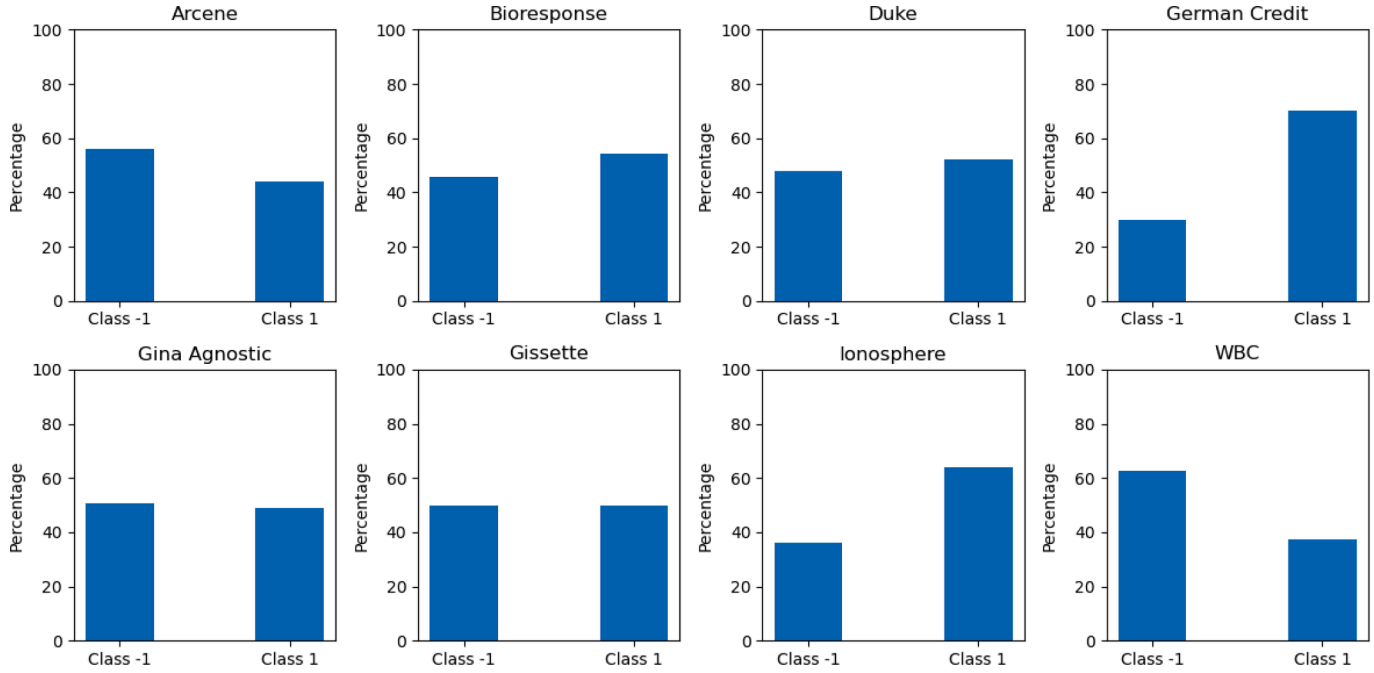


Fig. 4. Distribution of classes per instance.

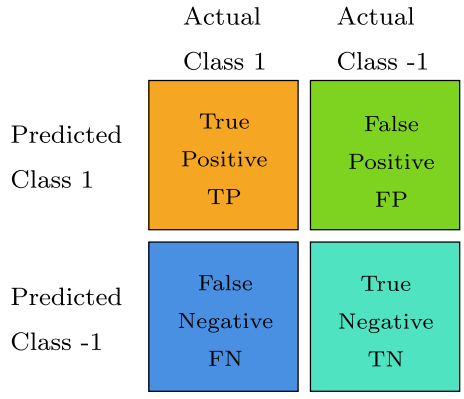


Fig. 5. Confusion matrix.

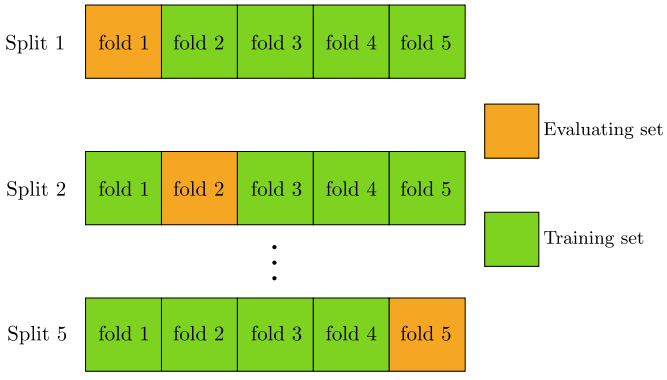


Fig. 6. Example K-fold cross-validation.

range of the initial features, respectively; w^l and w^u are lower and upper range of the initial weight, respectively; b^l and b^u are lower and upper range of initial intercept, respectively. Finally, ρ_0^1 and ρ_0^2 are the initial probability of mutation operators 1 and 2, respectively.

Given the stochastic nature of evolutionary algorithms and the extensive number of parameters involved, accurate parameter tuning is essential to achieve good computational performance. In this study, we utilized irace, an automatic configuration tool implemented in R [65], as an effective method for parameter tuning.

We included all instances in the parameter tuning. The tuning tool utilized the average results from three runs per instance, each with a different random seed, to explore various parameter configurations for the algorithm. For each instance evaluated by irace, the stopping condition was set to a time limit of 60 s, with the population size fixed at 50. Based on preliminary experiments, the population was fixed because we noticed that irace tended to select the highest possible value

Table 3
Set of parameter tuning of the proposed algorithm.

Parameter	Description	Value	NSGA-II$_1$	NSGA-II$_2$	NSGA-II$_3$
$\mathcal{K}$	population size	fixed	50	50	50
f^l	lower percentage range of the initial features	[0.01, 1.00]	0.027	0.028	0.598
f^u	upper percentage range of the initial features	[0.01, 1.00]	0.454	0.940	0.755
w^l	lower range of the initial weight	[−10.00, 10.00]	−6.877	−2.818	-9.460
w^u	upper range of the initial weight	[−10.00, 10.00]	4.684	3.511	6.873
b^l	lower range of initial intercept	[−10.00, 10.00]	−0.262	−5.131	−0.702
b^u	upper range of initial intercept	[−10.00, 10.00]	9.848	−3.283	1.306
ρ_0^1	initial probability of mutation operator 1	[0.00, 1.00]	0.535	0.665	0.672
ρ_0^2	initial probability of mutation operator 2	[0.00, 1.00]	0.812	0.011	0.563

Table 4
Comparison of results of the three versions considering the average of the five averages of the three metrics.

Dataset	Time	NSGA-II$_1$			NSGA-II$_2$			NSGA-II$_3$		
		AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
Arcene	1200	0.806	0.732	0.447	0.826	0.699	0.509	**0.838**	**0.769**	**0.542**
	3600	0.806	0.740	0.463	0.824	0.701	0.512	**0.841**	**0.765**	**0.540**
Bioresponse	1200	0.701	0.718	0.126	0.734	0.684	**0.360**	**0.756**	**0.737**	0.312
	3600	0.701	0.718	0.126	0.734	0.687	**0.363**	**0.757**	**0.737**	0.312
Duke	1200	0.983	**0.888**	**0.798**	**0.996**	0.728	0.567	0.974	0.862	0.700
	3600	0.981	**0.882**	**0.779**	**0.996**	0.735	0.578	0.970	0.865	0.711
German Credit	1200	0.539	**0.823**	0.000	**0.756**	0.747	**0.350**	0.540	**0.823**	0.000
	3600	0.539	**0.823**	0.000	**0.754**	0.747	**0.354**	0.540	**0.823**	0.000
Gina Agnostic	1200	0.848	0.787	0.555	0.848	0.711	0.502	**0.859**	**0.788**	**0.583**
	3600	0.847	**0.787**	0.556	0.847	0.711	0.502	**0.858**	**0.787**	**0.582**
Gisette	1200	0.869	0.809	0.572	**0.918**	0.734	0.617	0.882	**0.826**	**0.635**
	3600	0.888	0.822	0.610	**0.917**	0.734	0.618	0.889	**0.830**	**0.647**
Ionosphere	1200	**0.924**	**0.919**	**0.752**	0.920	0.901	0.725	0.913	0.917	0.745
	3600	**0.926**	**0.919**	**0.749**	0.920	0.902	0.729	0.913	0.917	0.743
WBC	1200	**0.996**	**0.969**	**0.949**	0.995	0.960	0.933	**0.996**	0.968	0.946
	3600	**0.996**	0.967	0.945	0.995	0.956	0.927	**0.996**	**0.968**	**0.946**
1200 avg.	–	0.833	0.831	0.525	0.874	0.770	0.570	0.845	0.836	0.558
3600 avg.	–	0.835	0.832	0.528	0.874	0.772	0.573	0.846	0.837	0.560
Hits	–	4	8	5	6	0	4	8	11	7

for this parameter, as larger values yielded better results. Additionally, these experiments indicated that the best parameter configurations vary depending on the evaluation metric. Consequently, we performed three separate parameter tuning based on the metrics AUC-ROC, F-Score, and CKC, resulting in the versions NSGA-II$_1$, NSGA-II$_2$, and NSGA-II$_3$, respectively. In each case, the objective was to minimize the metric value returned by our algorithm.

The first parameter tuning (NSGA-II$_1$) required approximately 53.793 h, while the second and third processes (NSGA-II$_2$ and NSGA-II$_3$) took approximately 57.994 and 63.510 h, respectively. All parameter tuning procedures were executed using 40 threads on the aforementioned supercomputing system. Table 3 presents the parameter names, descriptions, ranges of tested values, and the final values determined through irace tuning for each metric.

Although the three algorithms share parameter settings that seem similar thanks to the parameter tuning, they exhibit different search behaviors. NSGA-II$_1$ adopts an aggressive initial feature reduction strategy, generating solutions with at most 45.4 % of the features and employing high initial mutation probabilities of 81.2 % for adding or removing features and 53.5 % for swapping unselected features with selected ones. In contrast, NSGA-II$_2$ generates initial solutions of wide sizes, selecting up to 94 % of the features. It applies low mutation for adding or removing features (1.1 %) but prioritizes feature swapping (66.5 %), favoring feature exchange rather than elimination. Meanwhile, NSGA-II$_3$ is more conservative, favoring solutions with larger feature subsets, ranging between 59.8 % and 75.5 %, but compensates with aggressive mutations, with 67.2 % for adding/removing features and 56.3 % for swapping.

The algorithms also differ in their initial weight distributions. NSGA-II$_3$ allows a wide range of values, leaning toward negative weights, while NSGA-II$_1$ follows a similar trend but within a narrower range. NSGA-II$_2$, on the other hand, further restricts its weight range, incorporating fewer negative values. In terms of initial intercept values, NSGA-II$_1$ covers almost the entire positive range, with a small negative component, NSGA-II$_2$ generates only negative intercept values, and NSGA-II$_3$ concentrates on very small intercept values.

In summary, NSGA-II$_1$ is characterized by aggressive feature reduction and high mutation rates, NSGA-II$_2$ maintains diverse initial FS with low mutation for adding/removing features but high feature swapping, and NSGA-II$_3$ is more conservative in FS but aggressive in mutation rates. While all three algorithms follow the same structural framework, parameter tuning tailors each to a distinct search strategy, optimizing FS and mutation dynamics differently.

5.5. Performance results of the three versions of our algorithm

This section presents the computational experiment that evaluates the performance of three algorithm versions from parameter tuning: NSGA-II$_1$, NSGA-II$_2$, and NSGA-II$_3$. Three values, one for each different seed, are obtained for each of the five splits, from which the maximum and average are calculated. Using these five maximums and five averages, two types of results are computed for each metric: (i) the average of the five averages is reported in Table 4, and (ii) the average of the five maximums is presented in Table 5.

Both tables summarize the results for the three algorithm versions, which share the same structure. The first column lists the dataset names, while the second specifies the corresponding time limit. The subsequent three groups of columns represent the performance of NSGA-II$_1$, NSGA-II$_2$, and NSGA-II$_3$, respectively. Each group provides results for the metrics AUC-ROC, F-Score, and CKC. Finally, the antepenultimate row shows the average values for all metrics at the 1200-second time limit, the penultimate row presents the averages for the 3600-second time limit, and the last row corresponds to the number of hits indicating the instances where the best metric values were achieved.

NSGA-II$_3$ outperforms NSGA-II$_1$ and NSGA-II$_2$ across most datasets in the average of the five averages, particularly excelling in the F-Score metric, underscoring its ability to identify superior solutions as shown in Table 4. Specifically, NSGA-II$_3$ achieved the highest performance across the three metrics (AUC-ROC, F-Score, and CKC) in 8, 11, and 7 hits, respectively. This is followed by NSGA-II$_1$ with 4, 8, and 5 hits, respectively. Regarding average performance, NSGA-II$_3$ delivered the best results for F-Score across both time limits. Meanwhile, NSGA-II$_2$ achieved the highest averages for AUC-ROC and CKC at both time limits. Extending the time limit from 1200 to 3600 s produced marginal improvements across all algorithms, with the most noticeable enhancements observed in CKC. For example, in the Gisette dataset, NSGA-II$_3$ improved its CKC score from 0.635 to 0.647, highlighting the potential benefits of additional computing time in refining the solution quality.

NSGA-II$_3$ generally achieves competitive or superior performance compared to NSGA-II$_1$ and NSGA-II$_2$ in the average of the five maximums, particularly in the AUC-ROC metric. This behavior is shown in

Table 5
Comparison of results of the three versions considering the average of the five maximums of the three metrics.

Dataset	Time	NSGA-II$_1$			NSGA-II$_2$			NSGA-II$_3$		
		AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
Arcene	1200	0.849	0.780	0.528	0.878	0.801	**0.657**	**0.892**	**0.830**	0.646
	3600	0.849	0.786	0.543	0.878	0.806	**0.667**	**0.894**	**0.830**	0.645
Bioresponse	1200	0.779	0.742	0.271	0.758	0.716	0.420	**0.796**	**0.764**	**0.476**
	3600	0.779	0.742	0.272	0.758	0.724	0.422	**0.797**	**0.765**	**0.477**
Duke	1200	**1.000**	**1.000**	**1.000**	**1.000**	0.878	0.736	**1.000**	0.985	0.920
	3600	**1.000**	**1.000**	**1.000**	**1.000**	0.893	0.770	**1.000**	0.985	0.920
German Credit	1200	0.592	**0.844**	0.000	**0.810**	0.764	**0.412**	0.601	**0.844**	0.000
	3600	0.592	**0.844**	0.000	**0.807**	0.764	**0.412**	0.601	**0.844**	0.000
Gina Agnostic	1200	0.877	**0.816**	0.631	0.868	0.748	0.557	**0.882**	0.813	**0.636**
	3600	0.877	**0.816**	0.633	0.868	0.748	0.557	**0.882**	0.813	**0.637**
Gisette	1200	0.916	0.847	0.661	**0.932**	**0.878**	**0.757**	0.930	0.863	0.717
	3600	0.928	0.862	0.699	**0.932**	**0.878**	**0.758**	0.928	0.862	0.711
Ionosphere	1200	0.946	**0.945**	**0.819**	0.943	0.919	0.775	**0.954**	0.938	0.803
	3600	0.946	**0.945**	**0.816**	0.943	0.922	0.781	**0.954**	0.938	0.802
WBC	1200	**0.997**	0.975	**0.959**	0.996	0.968	0.945	**0.997**	**0.976**	0.958
	3600	**0.997**	0.975	**0.959**	0.996	0.968	0.945	**0.997**	**0.976**	0.958
1200 avg	–	0.869	0.869	0.609	0.898	0.834	0.658	0.882	0.877	0.645
3600 avg	–	0.871	0.871	0.615	0.898	0.838	0.664	0.882	0.877	0.644
Hits	–	4	8	6	6	2	6	12	8	4

Table 5. Specifically, NSGA-II$_3$ achieved the best performance across the two metrics of three metrics, AUC-ROC and F-Score, with 12 and 8, respectively. Meanwhile, in F-Score, NSGA-II$_3$ also achieved 8 hits, and NSGA-II$_2$ obtained the best hits in CKC with 6 hits. Regarding average performance, NSGA-II$_3$ delivered the best results for F-Score across both time limits. Meanwhile, NSGA-II$_2$ achieved the highest averages for AUC-ROC and CKC at both time limits. Regarding the impact of time limits, increasing the time limit to 3600 s results in minor improvements for most metrics across all algorithms, particularly in CKC.

Overall, while the results of the three algorithm variants are closely similar across both tables, NSGA-II$_3$ stands out as the most robust variant. It delivers superior performance across a greater number of datasets, achieving 26 hits in the average of the five averages and 24 hits in the average of the five maximums, particularly excelling in the AUC-ROC and F-Score metrics. In contrast, NSGA-II$_1$ obtained 17 hits in the average of the five averages and 18 hits in the average of the five maximums. Based on these results, NSGA-II$_3$ is selected as the best-performing variant from the parameter tuning.

5.6. Analysis of the three versions of our algorithm performance on the datasets

Our approach's performance seems to be influenced by its search behavior, initial FS strategy, and mutation dynamics across different datasets. This section analyzes the performance of the three versions of our algorithm on the datasets.

NSGA-II$_3$ excels in high-dimensional, large-sample datasets since it consistently outperforms NSGA-II$_1$ and NSGA-II$_2$ on datasets with high dimensionality, such as Arcene, Bioresponse, and Gisette, with d of 10,000; 1,776; 5,000, respectively. This can be attributed to its more conservative FS strategy, which prevents excessive feature elimination, thereby retaining informative variables crucial for classification in high-dimensional spaces. While NSGA-II$_1$ aggressively reduces features and NSGA-II$_2$ selects a wide range of feature subsets, NSGA-II$_3$ balances FS with aggressive mutations, allowing it to refine the feature set iteratively while preserving important information.

The performance of NSGA-II$_3$ is less dominant on small-sample datasets such as Duke, Ionosphere, and German Credit with n of 44, 351, and 1,000, respectively. Suggesting that more aggressive feature removal (as seen in NSGA-II$_1$) or high feature swapping (as seen in NSGA-II$_2$) might be more beneficial in small datasets where redundant features can harm generalization. Furthermore, NSGA-II$_3$'s aggressive mutation strategy may introduce unnecessary variability in smaller datasets, negatively affecting classification consistency.

On datasets with moderate dimensionality (d between 30 and 1000), NSGA-II$_3$ exhibits competitive but not always dominant performance. For instance, in Gina Agnostic ($d = 970$), NSGA-II$_3$ achieves the highest CKC (0.583), while NSGA-II$_1$ and NSGA-II$_2$ achieve lower scores. This suggests that its feature retention strategy works well when the dataset contains a moderate number of features, as it can balance the trade-off between removing redundant features and preserving useful information. However, in low-dimensional datasets like WBC ($d = 30$), where most features are already relevant, aggressive FS does not significantly impact performance, leading to comparable results among the three algorithms.

5.7. NSGA-II$_3$ against the state-of-the-art algorithm

This section presents the computational results of NSGA-II$_3$ against the state-of-the-art algorithm proposed by Valero-Carreras et al. [9], referred to here as NSGA-II$_V$. Three values are obtained for each of the five splits, from which the maximum and average are computed. These five maximums and five averages are then used to calculate two result types for each metric: (i) the average of the five averages and (ii) the average of the five maximums. The results are summarized in Table 6. Note that the results of NSGA-II$_V$ are directly obtained from Valero-Carreras et al. [9], where the parameter configurations and computational settings are detailed. Their algorithm was implemented in `C++`, and the experiments were conducted on a Scientific Computing Cluster running CentOS Linux 7.5.1804.

Table 6 presents the performance of the algorithms with the following structure: the first column lists the dataset names, and the second column specifies the corresponding time limit. The following two groups of columns display the performance of NSGA-II$_3$ and NSGA-II$_V$, respectively. Each group includes results for the metrics AUC-ROC, F-Score, and CKC based on the average of the five averages. Similarly, the subsequent two groups of columns follow the same format but report results based on the average of the five maximums. Finally, the antepenultimate row shows the average values for all metrics at the 1200-second time limit, the penultimate row presents the averages for the 3600-second

Table 6
Comparison of results of NSGA-II$_3$ against the state-of-the-art algorithm.

Dataset	Time	Average of the five averages						Average of the five maximums					
		NSGA-II$_3$			NSGA-II$_V$			NSGA-II$_3$			NSGA-II$_V$		
		AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
Arcene	1200	**0.838**	**0.769**	**0.542**	0.717	0.676	0.435	**0.892**	**0.830**	**0.646**	0.742	0.720	0.478
	3600	**0.841**	**0.765**	**0.540**	0.724	0.683	0.455	**0.894**	**0.830**	**0.645**	0.751	0.723	0.505
Bioresponse	1200	**0.756**	**0.737**	**0.312**	0.586	0.720	0.175	**0.796**	**0.764**	**0.476**	0.609	0.725	0.221
	3600	**0.757**	**0.737**	**0.312**	0.571	0.719	0.144	**0.797**	**0.765**	**0.477**	0.589	0.724	0.179
Duke	1200	**0.974**	**0.862**	**0.700**	0.617	0.623	0.236	**1.000**	**0.985**	**0.920**	0.627	0.627	0.255
	3600	**0.970**	**0.865**	**0.711**	0.555	0.558	0.112	**1.000**	**0.985**	**0.920**	0.629	0.623	0.257
German Credit	1200	0.540	**0.823**	0.000	**0.706**	0.593	**0.351**	0.601	**0.844**	0.000	**0.711**	0.597	**0.373**
	3600	0.540	**0.823**	0.000	**0.705**	0.591	**0.360**	0.601	**0.844**	0.000	**0.712**	0.599	**0.375**
Gina Agnostic	1200	**0.859**	**0.788**	**0.583**	0.611	0.678	0.221	**0.882**	**0.813**	**0.636**	0.619	0.679	0.238
	3600	**0.858**	**0.787**	**0.582**	0.650	0.683	0.300	**0.882**	**0.813**	**0.637**	0.688	0.694	0.378
Gisette	1200	**0.882**	**0.826**	**0.635**	0.684	0.758	0.368	**0.930**	**0.863**	**0.717**	0.699	0.769	0.399
	3600	**0.889**	**0.830**	**0.647**	0.711	0.766	0.423	**0.928**	**0.862**	**0.711**	0.773	0.797	0.545
Ionosphere	1200	**0.913**	**0.917**	**0.745**	0.858	0.915	0.742	**0.954**	**0.938**	**0.803**	0.858	0.916	0.744
	3600	**0.913**	0.917	0.743	0.863	**0.919**	**0.753**	**0.954**	**0.938**	**0.802**	0.866	0.923	0.762
WBC	1200	**0.996**	**0.968**	**0.946**	0.971	0.964	0.943	**0.997**	**0.976**	**0.958**	0.973	0.967	0.947
	3600	**0.996**	0.968	0.946	0.975	**0.969**	**0.950**	**0.997**	**0.976**	**0.958**	0.975	0.969	0.950
1200 avg	–	0.845	0.836	0.558	0.719	0.741	0.434	0.882	0.877	0.645	0.730	0.750	0.457
3600 avg	–	0.846	0.837	0.560	0.719	0.736	0.437	0.882	0.877	0.644	0.748	0.757	0.494
Hits	–	14	14	12	2	2	4	14	16	14	2	0	2

time limit, and the last row corresponds to the number of hits indicating the instances where the best metric values were achieved.

For the average of the five averages, NSGA-II$_3$ consistently outperforms the state-of-the-art algorithm across most datasets, achieving higher AUC-ROC, F-Score, and CKC values. Notable improvements are observed in datasets such as Duke and Gisette, where NSGA-II$_3$ demonstrates significant advantages in all metrics. In the average of the five maximums, the superiority of NSGA-II$_3$ becomes even more pronounced, with consistently higher values across nearly all datasets and metrics. The overall averages at the 1200-second time limit highlight NSGA-II$_3$'s superior performance, with an AUC-ROC of 0.882 compared to 0.730 for NSGA-II$_V$, an F-Score of 0.877 versus 0.750, and a CKC of 0.645 against 0.457. Similar trends are observed at the 3600-second time limit, where NSGA-II$_3$ maintains dominance. Regarding the hits row, NSGA-II$_3$ outperforms the baseline in all metrics, achieving 12 or more hits in AUC-ROC, F-Score, and CKC, compared to a maximum of four hits by the adversarial algorithm. In summary, NSGA-II$_3$ clearly and consistently outperforms the baseline algorithm in terms of average performance and peak values, reaffirming its effectiveness in addressing the problem at hand.

An exceptional situation can be observed in the case of the German Credit dataset, where the CKC values are zero for both time limits, while the AUC-ROC and F-Score remain within acceptable ranges. This outcome can be attributed to the strong class imbalance present in the dataset, which may lead the model to favor the majority class. While this behavior can still yield good performance in metrics such as F-Score, especially when the positive class dominates, the CKC penalizes lack of agreement beyond chance, resulting in lower scores. This highlights the value of including complementary metrics like CKC to capture different aspects of model behavior under challenging data distributions.

5.8. Statistic testing

To confirm the statistical significance of the observed differences in the results, we employed the critical difference diagrams [66], which are a graphical tool used to visualize and interpret the results of nonparametric statistical comparisons among multiple algorithms across multiple datasets for analyzing our experimental outcomes. These analyses focused on the average of the five averages and five maximums of the three metrics for all algorithms. Following the guidelines of Demšar [66], the critical difference diagrams consider the Friedman test [67] applied to evaluate the null hypothesis that all algorithms exhibit the same performance. Once we established a statistical difference within the algorithms' performance upon rejecting the null hypothesis, the pairwise posthoc analysis suggested by Benavoli et al. [68] is followed. Specifically, the Wilcoxon signed-rank test [69] was used along with Holm's alpha correction [70,71]. All tests were performed with a 95 % confidence level.

These diagrams plot the average ranks of algorithms, with lower ranks indicating better performance. Each algorithm is positioned on a horizontal axis according to its average rank. The critical difference value, calculated based on the number of algorithms, datasets, and the chosen significance level, is used to determine whether the observed differences are statistically significant. Algorithms that are not significantly different from each other are connected by a horizontal line. In contrast, if two algorithms are not connected, it means their performance difference is statistically significant.

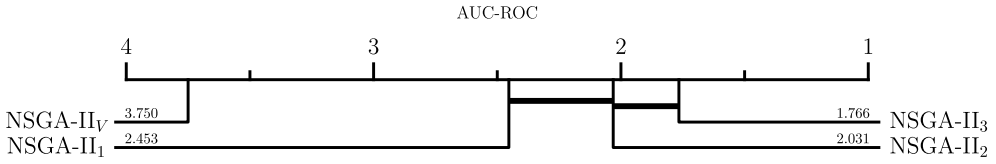


Fig. 7. Critical difference diagram of the dataset considering AUC-ROC.

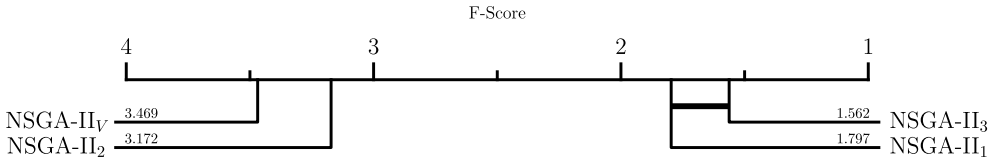


Fig. 8. Critical difference diagram of the dataset considering F-Score.

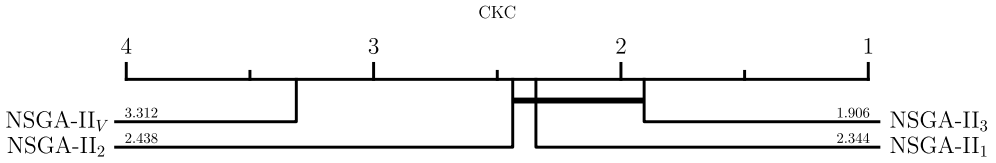


Fig. 9. Critical difference diagram of the dataset considering CKC.

Table 7
Results of ablation study with four modified versions of NSGA-II$_3$.

	NSGA-II$_3$			NSGA-II$_3^1$			NSGA-II$_3^2$			NSGA-II$_3^3$			NSGA-II$_3^4$		
	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
1200 avg	**0.845**	**0.836**	0.558	0.837	0.834	**0.561**	0.753	0.775	0.432	0.722	0.734	0.392	0.839	0.828	0.549
3600 avg	**0.846**	**0.837**	0.560	0.838	0.835	**0.562**	0.756	0.778	0.439	0.722	0.734	0.392	0.839	0.828	0.550
Hits	**12**	**10**	**6**	4	7	**6**	0	2	0	4	2	4	3	2	0
Friedman/Wilcoxon	4.525×10^{-6}	8.699×10^{-7}	2.237×10^{-4}	0.059	0.247	0.753	3.052×10^{-5}	0.001	0.001	0.002	1.526×10^{-4}	0.003	0.090	0.035	0.149

Figs. 7, 8, and 9 illustrate the critical difference diagrams, which show the performance rankings and statistically significant differences across algorithms for the three evaluation metrics (AUC-ROC, F-Score, and CKC, respectively). In particular, NSGA-II$_3$ consistently showed as the top-performing algorithm across all three metrics. Conversely, NSGA-II$_V$ consistently ranked last in the three metrics. This indicates that even considering the different search strategies of our approach, thanks to parameter tuning, the three versions of our algorithm show statistical differences with NSGA-II$_V$. However, some results indicated no statistically significant differences among our versions as in the AUC-ROC metric, where NSGA-II$_3$ showed no significant difference compared to NSGA-II$_2$, and NSGA-II$_2$ exhibited similar performance to NSGA-II$_1$. In the F-Score metric, NSGA-II$_3$ demonstrated no significant difference from NSGA-II$_1$. Finally, in the CKC metric, NSGA-II$_3$ showed no significant difference when compared to NSGA-II$_2$.

5.9. Ablation study

We conducted an ablation study [72] to evaluate the contribution of specific components within NSGA-II$_3$. This analysis compares the complete NSGA-II$_3$ against four modified versions, each with specific components either removed or simplified to assess their individual impact on performance. The configurations tested in the ablation study are as follows: (i) NSGA-II$_3^1$ has equal initial probabilities assigned to both mutation operators; (ii) NSGA-II$_3^2$ uses of only mutation operator 1; (iii) NSGA-II$_3^3$ uses of only mutation operator 2; and (iv) NSGA-II$_3^4$ has broader initialization ranges for features, weights, and intercepts.

All algorithms were executed under the experimental setup described in Section 5.3 to ensure a fair and consistent comparison. Table 7 summarizes the results of the complete NSGA-II$_3$ algorithm alongside its modified versions. We report performance for the average of the five averages for the eight datasets. The columns are grouped to represent the performance of NSGA-II$_3$, NSGA-II$_3^1$, NSGA-II$_3^2$, NSGA-II$_3^3$, and NSGA-II$_3^4$, respectively, with each group providing results for AUC-ROC, F-Score, and CKC. We report the average performance across the eight datasets at 1200 and 3600 s and the number of hits. In addition, we report the *p*-value of the Friedman test for the NSGA-II$_3$ group to determine whether the algorithms exhibit statistically significant differences in performance, shown only in the columns of NSGA-II$_3$. Meanwhile, in the columns of the modified versions, we provide the *p*-values of the Wilcoxon signed-rank test to evaluate whether the performance differences compared to NSGA-II$_3$ are statistically significant. Note that the used significance threshold was set at 0.05 for both nonparametric statistical tests.

Moreover, the details of the eight datasets in the ablation study are illustrated in Fig. 10. Each plot presents the results for one of the three evaluation metrics across all datasets, comparing the performance of our algorithm and its modified versions at 1200 and 3600 s. In the plots, each metric value for the algorithms is represented by a square, while dashed lines indicate the overall average performance of each version across all datasets. This visualization provides the superior performance of NSGA-II$_3$, mainly in AUC-ROC and F-Score metrics.

Overall, NSGA-II$_3$ outperforms its modified versions, particularly in the AUC-ROC and F-Score metrics at 1200 and 3600 s and in the number of hits. However, NSGA-II$_3^1$ achieves better performance in the CKC metric, indicating that certain configurations may favor specific evaluation criteria. The Friedman test rejects the null hypothesis that all algorithms perform equivalently, suggesting the presence of statistically significant differences among them. Therefore, we applied the Wilcoxon signed-rank test for pairwise comparisons between NSGA-II$_3$ and each modified version. The results of the Wilcoxon test show that NSGA-II$_3$ exhibits statistically significant improvements over NSGA-II$_3^2$ and NSGA-II$_3^3$ across all three metrics. When compared with NSGA-II$_3^4$, the difference is significant only for the AUC-ROC metric. In contrast, no statistically significant difference exists between NSGA-II$_3$ and NSGA-II$_3^1$ in any of the three metrics.

The ablation study highlights both mutation operators' critical role in our algorithm's effectiveness. When either mutation operator is removed, as in NSGA-II$_3^2$ and NSGA-II$_3^3$, we observe a noticeable decline in performance across all metrics. This suggests that each operator contributes uniquely to the algorithm's ability to explore and exploit the solution space. In contrast, configurations that retain both mutation operators, such as NSGA-II$_3^1$, achieve performance levels that are comparable to the original NSGA-II$_3$, reinforcing the importance of their combined use. Furthermore, while the initialization ranges for features, weights, and intercepts also influence performance, their impact appears to be less significant than that of the mutation strategy. For instance, NSGA-II$_3^4$, which uses broader initialization ranges, performs

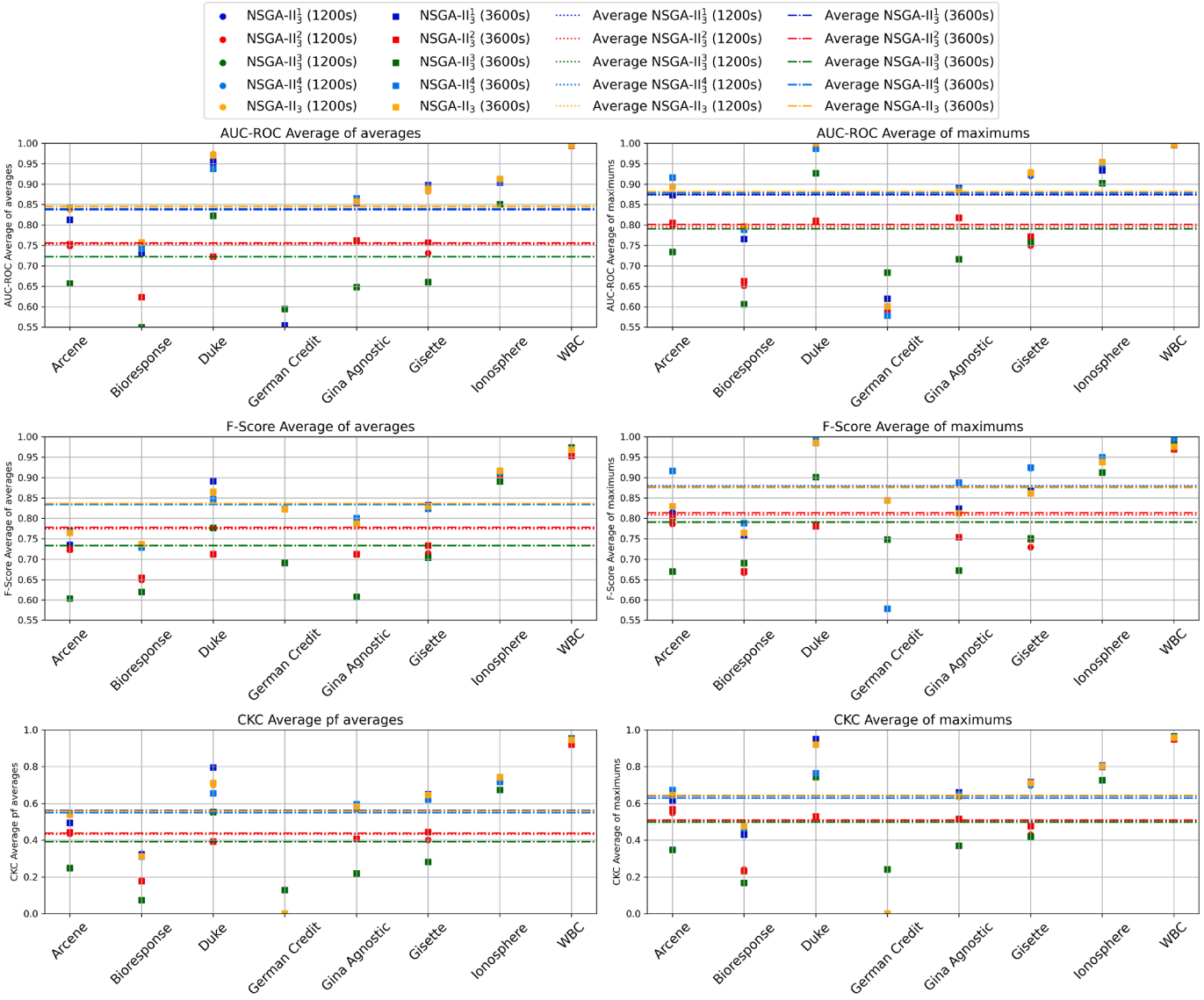


Fig. 10. Metrics comparison in ablation study.

only slightly worse than NSGA-II$_3$, indicating that although initialization helps guide early search behavior, it does not compensate for the diversity and adaptability provided by the tailored mutation operators.

5.10. Algorithm convergence and scalability

To evaluate the algorithm's convergence over iterations, the binary cross-entropy loss function [11] was utilized. This metric effectively demonstrates the SVM's predictive performance as the NSGA-II progresses, providing a way to illustrate its convergence within the context of a multi-objective problem. To adapt the function to the context of a multi-objective SVM, we used the average of $\mathcal{L}$, as defined in Eq. (24), computed over each solution in the first Pareto front. In this equation, y_i' denotes the label value converted to 0/1, and $p(y_i)$ is the probability using a sigmoid function from SVM outputs. The binary cross-entropy penalizes incorrect high-probability predictions, reflecting the algorithm's progress in certainty and predictive accuracy.

$$\mathcal{L} = -\frac{1}{|N|}\sum_{i\in N}[y_i' \log(p(y_i)) + (1-y_i')\log(1-p(y_i))] \tag{24}$$

Fig. 11 presents the loss function plots for each evaluated instance, with an iteration limit based on the algorithm convergence of the last dataset (Gisette). Across all datasets, the loss function consistently decreases, reflecting the algorithm's continuous improvement and validating the reported metrics. Note that all instances converged within 10 min, as shown on the secondary time axis in the graphs. We can observe that the number of features influences convergence time, explaining the slower convergence of Gisette (5,000 features) compared to smaller datasets like Ionosphere and WBC (34 and 30 features, respectively). This rapid convergence underscores the algorithm's efficiency and robustness. Additionally, a zoomed-in view of the datasets is provided to better visualize the details of their convergence behavior.

To analyze the algorithm's computational scalability concerning dataset size, we conducted a computing time experiment using a fixed stopping criterion of 1000 iterations per dataset, rather than a given time limit. The results are summarized in Fig. 12, which presents the average computation time (in seconds) required for each dataset, annotated with the number of features and samples per dataset directly on the bars. As shown in the figure, computation time generally increases with dataset size, particularly as the number of features and observations grows. Notably, datasets like Gisette (5,000 features, 13,500 samples) and Arcene (10,000 features) result in significantly higher computing time compared to smaller datasets like WBC or German Credit. This trend confirms the algorithm's expected behavior as complexity

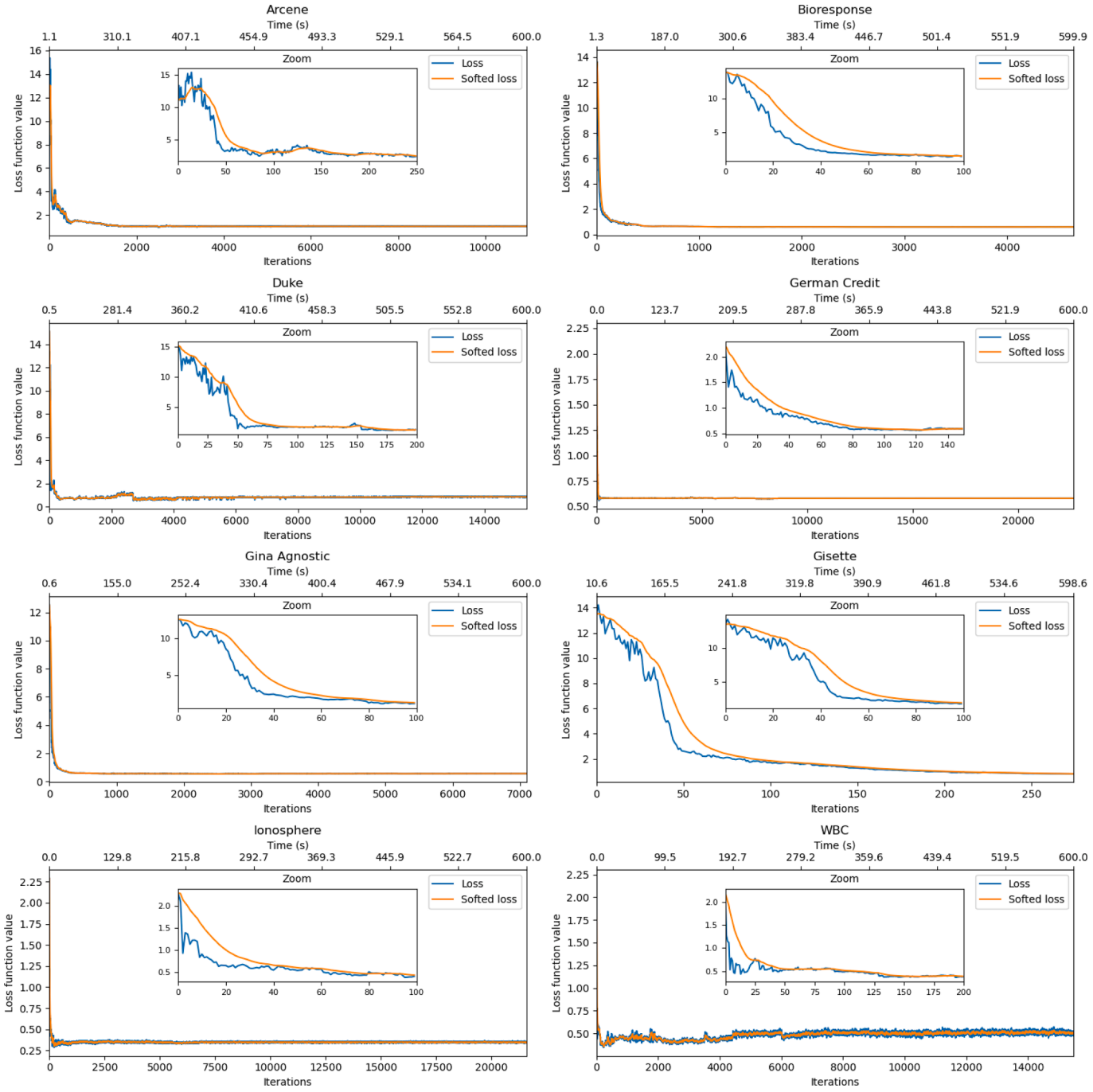


Fig. 11. A sample of loss function for each dataset.

increases. Furthermore, this plot of computing time versus dataset size provides a clear visual illustration of the algorithm's scalability. It also highlights the algorithm's efficiency across a wide range of dataset sizes. Since our approach is designed for feature selection, the reduction in the number of active features during optimization further helps to mitigate computational load, especially in high-dimensional datasets where fewer relevant features are ultimately retained.

5.11. *Performance of NSGA-II$_3$ on large-scale datasets*

To evaluate the scalability and robustness of our proposed approach, we tested NSGA-II$_3$ on five real-world large-scale datasets, ranging in size from 101,763 to 5,000,000 samples. The binary classification with a software defects dataset [73] (Software) is derived from industrial software systems and is used for predicting defect-prone modules. The Santander customer transaction prediction dataset [74] (Santander) focuses on a financial use case, aiming to classify customer behavior based on anonymized transaction data. The credit card fraud detection dataset [75] (Fraud) is notable for its extreme class imbalance, where fraudulent transactions represent only 0.172% of the data. The Higgs boson machine learning challenge dataset [76] (Higgs) aims to differentiate between signal and background events in particle physics. Lastly, the supersymmetry (SUSY) dataset [77] simulates events from high-energy physics, making it the largest and one of the most complex datasets in our experiments. All large-scale datasets used can be accessed on the Kaggle platform [58]. Table 8 summarizes the performance of

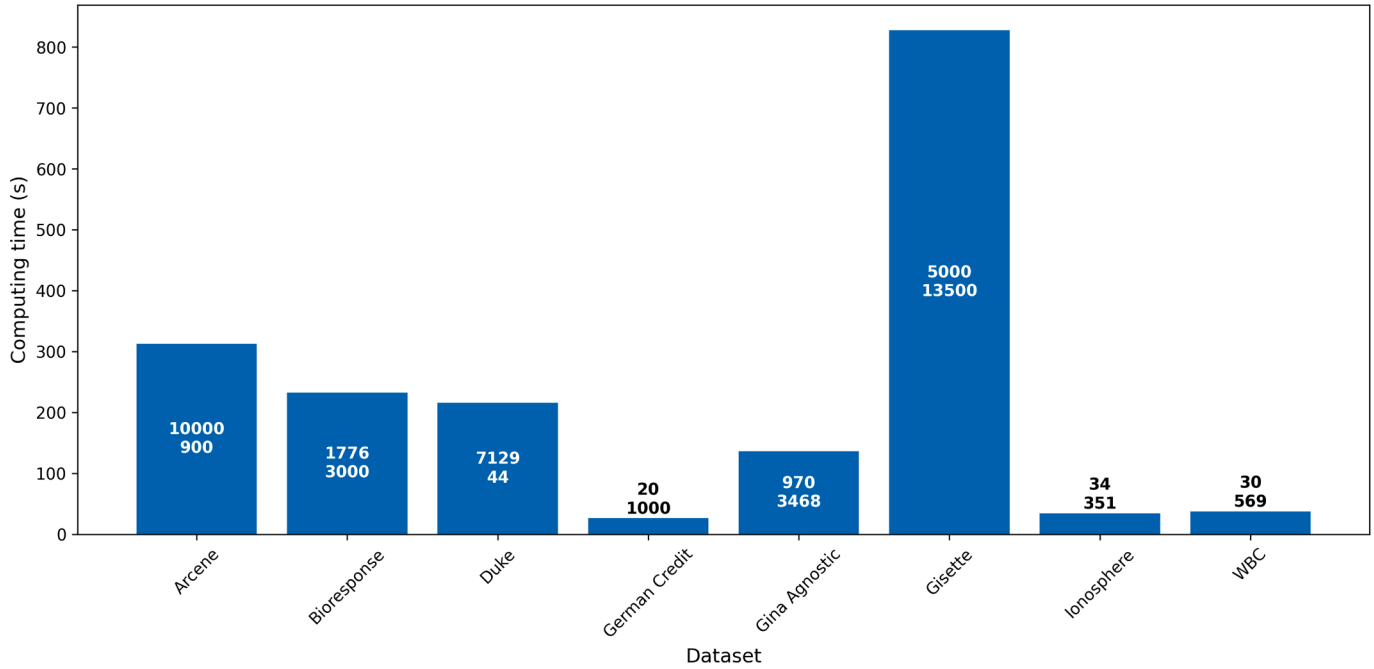


Fig. 12. Average computation time (in seconds) for each dataset after 1000 iterations.

Table 8
Results of NSGA-II$_3$ on large-scale datasets.

Dataset	n	d	Comparison				NSGA-II$_3$		
			Reference	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
Software	101,763	20	Kim and Kim [73]	0.736	0.717	–	0.751	0.874	0.018
Santander	200,000	200	Mohammed et al [74]	0.998	0.998	–	0.646	0.227	0.072
Fraud	287,807	30	Awoyemi et al. [75]	–	0.1608	0.1581	0.981	0.009	0.006
Higgs	800,000	30	Azhari et al. [76]	0.796	–	–	0.566	0.434	0.094
SUSY	5,000,000	18	Baldi et al. [77]	0.885	–	–	0.832	0.731	0.527

NSGA-II$_3$ compared to reference results reported in prior studies based on the AUC-ROC, F-Score, and CKC metrics, considering the average of the five maximums.

NSGA-II$_3$ achieved promising results in several of the tested datasets, particularly in the Software dataset, where it outperformed the baseline AUC-ROC and F-Score values reported by Kim and Kim [73], improving from 0.736 to 0.751 and from 0.717 to 0.874, respectively. This indicates that our algorithm is capable of delivering both high predictive accuracy and robustness in real-world, high-volume industrial data settings. However, the results were mixed across the remaining datasets. For instance, in the Fraud dataset, NSGA-II$_3$ achieved an AUC-ROC of 0.981. Nevertheless, its F-Score and CKC were relatively low (0.009 and 0.006) compared to Awoyemi et al. [75], which can be attributed to the high class imbalance inherent in this dataset, highlighting a potential limitation in the current weighting strategy.

In the Santander and Higgs datasets, NSGA-II$_3$ showed lower AUC-ROC and F-Score values than the references. These results suggest that while the algorithm is efficient for moderate feature spaces and specific real-world settings, its performance can be impacted by extremely high data volumes and highly competitive classification environments, mainly where complex deep learning models are often used. On the SUSY dataset, which contains 5 million instances, NSGA-II$_3$ delivered a strong performance, achieving an AUC-ROC of 0.832 and a CKC of 0.527, close to the reference AUC-ROC of 0.885 from Baldi et al. [77]. This demonstrates that NSGA-II$_3$ can scale effectively even in massive datasets with relatively compact feature space. In summary, NSGA-II$_3$ exhibits competitive performance on large-scale datasets, especially where FS and interpretability are prioritized. However, its effectiveness can vary depending on the degree of class imbalance and the complexity of the decision boundaries.

5.12. Discussion

The solution representation proposed in this paper, which directly generates the weights, allows a straightforward definition of hyperplanes that partition the solution space. This approach differs from Valero-Carreras et al. [9], where hyperplanes are computed by selecting vectors from the dataset and determining their intersections, as described in Alcaraz et al. [8]. Recently, Alcaraz [10] highlighted the limitations of such representations, noting that they constrain the search space by excluding many potential solutions. Regarding recent encoding in Alcaraz [10], our representation remains more straightforward, as this latter encoding still computes the intersecting hyperplane but with a reduced number of support vectors, imposing only one per class. Thus, our proposed representation offers a significant advantage by reducing the computing time per individual, allowing for a higher number of iterations. As a result, the algorithm can explore the solution space more extensively, potentially improving overall performance. Notably, this benefit is achieved despite the computational implementation of Valero-Carreras et al. [9] since the algorithm is implemented in `C++`, a language that can be up to approximately 70 times faster than Python for specific tasks [78].

The algorithm proposed in this work dynamically adjusts the number of features considered during execution in an unsupervised manner, aiming to converge to a good feature set. This approach differs from Valero-Carreras et al. [9], which employs a static selection of five features. Our algorithm achieves this adaptability by modifying constraint

(12), replacing the fixed feature parameter with a dynamic constraint based on an upper bound determined by the total number of features. This flexibility enhances the algorithm's ability to explore the search space more effectively, allowing it to identify the most informative features and improve predictive performance.

Including an imbalance penalty in the dataset was crucial in addressing class imbalances. By incorporating the parameter α_i into the objective function, the importance of each misclassified vector is weighted according to its class prevalence relative to the total number of vectors in the dataset. This adjustment ensures the algorithm emphasizes accurate predictions for the predominant class, particularly in datasets with significant imbalances. This approach allowed the population of solutions to adapt more effectively to the data distribution, enhancing the classification performance for the underrepresented class while maintaining overall predictive accuracy. As a result, the metrics used to evaluate performance, such as the AUC-ROC and F-Score, show significant improvements. These metrics highlight the algorithm's capability to achieve a balanced trade-off between precision and recall, making it well-suited for applications where class imbalance poses a critical challenge.

Regarding the CKC metric, both our algorithm and the algorithm proposed by Valero-Carreras et al. [9] exhibit relatively low values for this metric. This study has included it primarily to enable a direct comparison between the two algorithms. However, it is essential to acknowledge the criticisms of CKC, as highlighted by Feinstein and Cicchetti [79], particularly its susceptibility to imbalances in the marginal totals of the confusion matrix. This susceptibility can result in low CKC values, even when the actual agreement between predicted and true values is high. The calculation of CKC, shown in Eq. (23), depends on the confusion matrix's values. The numerator of p_c includes the product of marginal probabilities for correct and incorrect classifications. This structure makes the metric highly sensitive to class imbalances, which can skew the perceived agreement. Consequently, CKC may underestimate the predictive performance of a model in cases where other metrics, such as the AUC-ROC or F-Score, validate its strong predictive capability. In this context, while CKC provides a standard measure for comparison, its limitations emphasize the importance of relying on complementary metrics, particularly in scenarios involving imbalanced datasets. Metrics like AUC-ROC and F-Score offer a more nuanced understanding of the algorithm's performance, effectively capturing its ability to balance precision and recall.

Regarding computational resource requirements, all experiments were conducted in a CPU-based environment, without the need for specialized GPU acceleration. The proposed NSGA-II$_3$ algorithm was designed to be computationally efficient and memory-conscious, handling datasets with up to several million samples while maintaining memory usage below 8 GB of RAM. Although the current implementation is sequential, the algorithm's structure, particularly the evaluation of individuals and non-dominated sorting, is well-suited for parallelization.

An additional practical insight arising from our experiments is the differentiated behavior of the three NSGA-II variants depending on dataset characteristics. Specifically, NSGA-II$_1$ generally excels in smaller or well-separated datasets, offering faster convergence with fewer selected features. In contrast, NSGA-II$_2$ performs more robustly in high-dimensional or noisy environments due to its enhanced feature filtering capacity. Finally, NSGA-II$_3$ shows stronger performance in imbalanced or uncertain datasets due to its balanced weighting and enhanced stability. While these guidelines are not prescriptive, these empirical trends provide preliminary heuristics for researchers and practitioners to guide the selection of the most suitable variant in their specific problem context.

6. Conclusions

This paper proposes a multi-objective metaheuristic integrated with the SVM and FS, specifically NSGA-II, to approximate the Pareto-optimal frontier while simultaneously optimizing the two SVM objectives and selecting features. Our approach leverages the soft-margin SVM model with integrated FS, allowing flexibility in the number of selected features. The proposed algorithm introduces a novel individual representation, along with specialized crossover and mutation operators designed to maintain dominance principles in multi-objective optimization. A weighted optimization strategy is also implemented to mitigate dataset imbalances.

Additionally, due to the parameter tuning based on the three performance metrics, generating three versions of our approach exhibits distinct search behaviors. The first version shows aggressive feature reduction with high mutation rates. The second version shows a diverse initial FS, with low mutation for adding/removing features and high feature swapping. The last version shows conservative FS, maintaining larger feature subsets but high mutation rates. Based on the ablation study, we find that both mutation operators contribute significantly to the algorithm's performance, with their combined use leading to more effective exploration and convergence. In contrast, initialization strategies, while helpful in shaping the early search process, have a comparatively smaller impact on overall optimization quality.

Three versions of our algorithm were evaluated on well-known binary classification datasets. The results show that three versions outperform the state-of-the-art algorithm in both solution quality and predictive performance within the given time limits. Highlighting the third version showed the best results among the three versions. The improvements in representation, tailored genetic operators, weighted optimization strategy, and effective parameter tuning guided by different evaluation metrics further refine performance, allowing for faster convergence without sacrificing classification accuracy.

When analyzing how the method scales with increasing feature dimensionality, NSGA-II$_3$ demonstrates strong performance on high-dimensional datasets. Arcene (10,000 features) consistently achieves the highest AUC-ROC, F-Score, and CKC across both time limits. Similarly, on Gisette (5,000 features), it maintains competitive results, particularly in F-Score and CKC, outperforming or closely matching the other variants. In contrast, the performance gap between the variants narrows for lower-dimensional datasets such as WBC and German Credit, where most features are already informative. These results highlight that NSGA-II$_3$ is particularly well-suited for high-dimensional feature spaces, where effective selection and robust mutation strategies are key to maintaining classification performance.

When analyzing how the method scales with datasets of increasing sample size, NSGA-II$_3$ demonstrates competitive performance, particularly in scenarios where FS and interpretability are key priorities. However, its effectiveness can vary depending on factors such as class imbalance and the complexity of decision boundaries. In datasets with extreme imbalance or highly nonlinear relationships, performance may be limited by the algorithm's current design. These observations point to promising directions for future work, including integrating adaptive sampling strategies or deep learning components to enhance robustness in challenging large-scale scenarios.

Several avenues can be explored for future research to enhance the algorithm's capabilities further:

- Introducing local exploration mechanisms when the algorithm reaches convergence could help refine solutions and prevent premature stagnation. In this regard, the integration of alternative diversity preservation strategies could also contribute to improving the exploration of the solution space and promoting a better distribution of solutions along the Pareto front. Furthermore, analyzing the shape and distribution of the Pareto front could offer valuable insights into the trade-offs between structural and empirical error, as well as the diversity of the solutions.
- Additional performance metrics, such as accuracy or cross-entropy, could be integrated to provide a more comprehensive evaluation and facilitate comparisons with other studies [80].

- Extending the current bi-objective algorithm into a tri-objective formulation by incorporating an additional objective, such as a loss function (e.g., cross-entropy), or one of the mentioned metrics, which could directly enhance performance by explicitly targeting the desired outcomes.
- A parallelized implementation of key operations (e.g., fitness evaluation, sorting) and leveraging GPU acceleration could significantly improve computational efficiency, making the approach even more suitable for very large-scale datasets.
- Exploring uncertainty quantification techniques to assess the sensitivity of model performance to input perturbations and enhance robustness analysis, especially considering varying degrees of class imbalance through controlled experiments.
- Exploring the effect of using alternative genetic operators in the mutation, crossover, and selection to understand convergence behavior better and improve optimization performance.
- Exploring the use of transfer learning or pretraining to enhance convergence and efficiency, especially in high-dimensional scenarios.
- Incorporating visual and quantitative comparisons of Pareto fronts between the proposed method and baseline approaches would provide an additional perspective on the behavior of the multi-objective optimization process, complementing the confusion-matrix-based evaluation adopted in this study.
- Another possible extension is the development of metaheuristics, such as the multi-objective evolutionary algorithm based on decomposition [81] and NSGA-III [82], based on the considered model, and incorporating the above ideas.

These future directions aim to refine the algorithm's performance, ensure robustness across diverse datasets, and enable more meaningful comparisons with alternative methods.

CRediT authorship contribution statement

Mathias Badilla-Salamanca: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Rosa Medina Durán:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Carlos Contreras-Bolton:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Data availability

We have shared the link to our data/code in the manuscript.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Carlos Contreras-Bolton reports financial support was provided by National Agency for Research and Development. Carlos Contreras-Bolton reports financial support was provided by National Laboratory High Performance Computing. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research was partially supported by the supercomputing infrastructure of the NLHPC (CCSS210001) and also funded by ANID (National Agency for Research and Development) for its support through the FONDECYT Iniciación Project number 11241132. The authors would like to express their gratitude to the anonymous reviewers for their valuable comments and suggestions, which helped to improve our work.

References

[1] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297. https://doi.org/10.1023/A:1022627411411
[2] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, A. Lopez, A comprehensive survey on support vector machine classification: applications, challenges and trends, Neurocomputing 408 (2020) 189–215. https://doi.org/10.1016/j.neucom.2019.10.118
[3] R.E. Bellman, Adaptive Control Processes: A Guided Tour, Princeton University Press, Princeton, 1961. https://doi.org/10.1515/9781400874668
[4] G.V. Trunk, A problem of dimensionality: a simple example, IEEE Trans. Pattern Anal. Mach. Intell. 1 (3) (1979) 306–307. https://doi.org/10.1109/TPAMI.1979.4766926
[5] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Elect. Eng. 40 (1) (2014) 16–28. https://doi.org/10.1016/j.compeleceng.2013.11.024
[6] J. Miao, L. Niu, A survey on feature selection, Procedia. Comput. Sci. 91 (2016) 919–926. https://doi.org/10.1016/j.procs.2016.07.111
[7] D. Theng, K.K. Bhoyar, Feature selection techniques for machine learning: a survey of more than two decades of research, Knowl. Inf. Syst. 66 (3) (2024) 1575–1637. https://doi.org/10.1007/s10115-023-02010-5
[8] J. Alcaraz, M. Labbé, M. Landete, Support vector machine with feature selection: a multiobjective approach, Expert Syst. Appl. 204 (2022) 117485. https://doi.org/10.1016/j.eswa.2022.117485
[9] D. Valero-Carreras, J. Alcaraz, M. Landete, Comparing two SVM models through different metrics based on the confusion matrix, Comput. Operat. Res. 152 (2023) 106131. https://doi.org/10.1016/j.cor.2022.106131
[10] J. Alcaraz, Redesigning a NSGA-II metaheuristic for the bi-objective support vector machine with feature selection, Comput. Oper. Res. 172 (2024) 106821. https://doi.org/10.1016/j.cor.2024.106821
[11] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer, Berlin, Heidelberg, 1 edition, 2007.
[12] S. Maldonado, J. Pérez, R. Weber, M. Labbé, Feature selection for support vector machines via mixed integer linear programming, Inf. Sci. 279 (2014) 163–175. https://doi.org/10.1016/j.ins.2014.03.110
[13] H. Aytug, Feature selection for support vector machines using generalized benders decomposition, Eur. J. Oper. Res. 244 (1) (2015) 210–218. https://doi.org/10.1016/j.ejor.2015.01.006
[14] M. Gaudioso, E. Gorgone, M. Labbé, A.M. Rodríguez-Chía, Lagrangian relaxation for SVM feature selection, Comput. Oper. Res. 87 (2017) 137–145. https://doi.org/10.1016/j.cor.2017.06.001
[15] S. Benítez-Peña, R. Blanquero, E. Carrizosa, P. Ramírez-Cobo, Cost-sensitive feature selection for support vector machines, Comput. Oper. Res. 106 (2019) 169–178. https://doi.org/10.1016/j.cor.2018.03.005
[16] M. Labbé, L.I. Martínez-Merino, A.M. Rodríguez-Chía, Mixed integer linear programming for feature selection in support vector machine, Discrete Appl. Math. 261 (2019) 276–304. https://doi.org/10.1016/j.dam.2018.10.025
[17] I.G. Lee, Q. Zhang, S.W. Yoon, D. Won, A mixed integer linear programming support vector machine for cost-effective feature selection, Knowl. Based Syst. 203 (2020) 106145. https://doi.org/10.1016/j.knosys.2020.106145
[18] M. Baldomero-Naranjo, L.I. Martínez-Merino, A.M. Rodríguez-Chía, A robust SVM-based approach with feature selection and outliers detection for classification problems, Expert Syst. Appl. 178 (2021) 115017. https://doi.org/10.1016/j.eswa.2021.115017
[19] G. Mavrotas, K. Florios, An improved version of the augmented ϵ-constraint method (AUGMECON2) for finding the exact pareto set in multi-objective integer programming problems, Appl. Math. Comput. 219 (18) (2013) 9652–9669. https://doi.org/10.1016/j.amc.2013.03.002
[20] A. Bouraoui, S. Jamoussi, Y. BenAyed, A multi-objective genetic algorithm for simultaneous model and feature selection for support vector machines, Artif. Intell. Rev. 50 (2) (2018) 261–281. https://doi.org/10.1007/s10462-017-9543-9
[21] H. Faris, M.A. Hassonah, A.M. Al-Zoubi, S. Mirjalili, I. Aljarah, A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a robust system architecture, Neural Comput. Appl. 30 (8) (2018) 2355–2369. https://doi.org/10.1007/s00521-016-2818-2
[22] A. Candelieri, I. Giordani, F. Archetti, K. Barkalov, I. Meyerov, A. Polovinkin, A. Sysoyev, N. Zolotykh, Tuning hyperparameters of a SVM-based water demand forecasting system through parallel global optimization, Comput. Oper. Res. 106 (2019) 202–209. https://doi.org/10.1016/j.cor.2018.01.013
[23] M. Sabzekar, Z. Aydin, A noise-aware feature selection approach for classification, Soft Comput. 25 (8) (2021) 6391–6400. https://doi.org/10.1007/s00500-021-05630-7
[24] W. Dudzik, J. Nalepa, M. Kawulok, Evolving data-adaptive support vector machines for binary classification, Knowl. Based Syst. 227 (2021) 107221. https://doi.org/10.1016/j.knosys.2021.107221
[25] T. Huang, C.-K. Ngan, Y.T. Cheung, M. Marcotte, B. Cabrera, A hybrid deep learning-based feature selection approach for supporting early detection of long-term behavioral outcomes in survivors of cancer: cross-sectional study, JMIR Bioinform. Biotechnol. 6 (2025). https://doi.org/10.2196/65001
[26] S. Abasabadi, H. Nematzadeh, H. Motameni, E. Akbari, Hybrid feature selection based on SLI and genetic algorithm for microarray datasets, J. Supercomput. 78 (18) (2022) 19725–19753. https://doi.org/10.1007/s11227-022-04650-w

[27] A. Jain, V. Jain, Sentiment classification using hybrid feature selection and ensemble classifier, J. Intell. Fuzzy Syst. 42 (2) (2022) 659–668. https://doi.org/10.3233/JIFS-189738

[28] J. Wang, X. Wang, X. Li, J. Yi, A hybrid particle swarm optimization algorithm with dynamic adjustment of inertia weight based on a new feature selection method to optimize SVM parameters, Entropy 25 (3) (2023). https://doi.org/10.3390/e25030531

[29] I. Bomze, F. D'Onofrio, L. Palagi, B. Peng, Feature selection in linear support vector machines via a hard cardinality constraint: a scalable conic decomposition approach, 2025. https://doi.org/10.1016/j.ejor.2025.03.017

[30] A. Zandvakili, M.M. Javidi, N. Mansouri, Simultaneous feature selection and SVM optimization based on fuzzy signature and chaos goa, Evol. Syst. 15 (5) (2024) 1907–1937. https://doi.org/10.1007/s12530-024-09595-4

[31] S. Afreen, A.K. Bhurjee, R.M. Aziz, Feature selection using game shapley improved grey wolf optimizer for optimizing cancer classification, Knowl. Inf. Syst. (2025) 1–32. https://doi.org/10.1007/s10115-025-02340-6

[32] Y. Xue, C. Zhang, F. Neri, M. Gabbouj, Y. Zhang, An external attention-based feature ranker for large-scale feature selection, Knowl. Based Syst. 281 (2023) 111084. https://doi.org/10.1016/j.knosys.2023.111084

[33] Y. Xue, C. Zhang, A novel importance-guided particle swarm optimization based on MLP for solving large-scale feature selection problems, Swarm Evol. Comput. 91 (2024) 101760. https://doi.org/10.1016/j.swevo.2024.101760

[34] F. Farokhmanesh, M.T. Sadeghi, Deep neural networks regularization using a combination of sparsity inducing feature selection methods, Neural Process. Lett. 53 (1) (2021) 701–720. https://doi.org/10.1007/s11063-020-10389-3

[35] L. Bote-Curiel, S. Ruiz-Llorente, S. Muñoz-Romero, M. Yagüe-Fernández, A. Barquín, J. García-Donas, J.L. Rojo-álvarez, Multivariate feature selection and autoencoder embeddings of ovarian cancer clinical and genetic data, Expert Syst. Appl. 206 (2022) 117865. https://doi.org/10.1016/j.eswa.2022.117865

[36] H.H.S. Junaid, F. Daneshfar, M.A. Mohammad, Automatic colorectal cancer detection using machine learning and deep learning based on feature selection in histopathological images, Biomed. Signal Process. Control 107 (2025) 107866. https://doi.org/10.1016/j.bspc.2025.107866

[37] Y. Bouchlaghem, Y. Akhiat, K. Touchanti, S. Amjad, A novel feature selection method with transition similarity measure using reinforcement learning, Decision Anal. J. 11 (2024) 100477. https://doi.org/10.1016/j.dajour.2024.100477

[38] V.N. Vapnik, The nature of statistical learning theory, Springer-Verlag, Berlin, Heidelberg, 1995.

[39] V. Vapnik, A. Chervonenkis, A note on one class of perceptions, Autom. Remote Control 25 (1964) 416. The original article is in Russian.

[40] V. Vapnik, A. Chervonenkis, Theory of Pattern Recognition, Nauka, Moscow, 1974, The original article is in Russian.

[41] M. Sabzekar, H.S. Yazdi, M. Naghibzadeh, Relaxed constraints support vector machine, Expert Syst. 29 (5) (2012) 506–525. https://doi.org/10.1111/j.1468-0394.2011.00611.x

[42] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction: Foundations and Applications, Springer-Verlag, Berlin, Heidelberg, 2006. https://doi.org/10.1007/978-3-540-35488-8

[43] J. Li, K. Cheng, S. Wang, F. Morstatter, R.P. Trevino, J. Tang, H. Liu, Feature selection: a data perspective, ACM Comput. Surv. 50 (6) (2017). https://doi.org/10.1145/3136625

[44] X. Yang, Q. Song, A. Cao, Weighted support vector machine for data classification, in: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., volume 2, 2005, pp. 859–864. https://doi.org/10.1109/IJCNN.2005.1555965

[45] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer Berlin Heidelberg, 2nd edition, 2015. https://doi.org/10.1007/978-3-662-44874-8

[46] J. Malczewski, C. Rinner, Multiobjective Optimization Methods, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 123–143. https://doi.org/10.1007/978-3-540-74757-4_5

[47] D.E. Goldberg, R. Lingle, Alleles, Loci and the Traveling Salesman Problem, in: Proceedings of the 1st International Conference on Genetic Algorithms, L. Erlbaum Associates Inc., USA, 1985, pp. 154–159. https://doi.org/10.5555/645511.657095

[48] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680. https://doi.org/10.1126/science.220.4598.671

[49] I. Guyon, S. Gunn, A. Ben-Hur, G. Dror, Arcene, 2008, (UCI Machine Learning Repository). https://doi.org/10.24432/C58P55

[50] T. Abdunabi, O. Basir, Predicting a biological response of molecules from their chemical properties using diverse and optimized ensembles of stochastic gradient boosting machine, in: 2014 IEEE International Conference on Big Data (Big Data), 2014, pp. 10–17. https://doi.org/10.1109/BigData.2014.7004386

[51] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson, J.R. Marks, J.R. Nevins, Predicting the clinical status of human breast cancer by using gene expression profiles, Proceedings of the National Academy of Sciences 98 (20) (2001) 11462–11467. https://doi.org/10.1073/pnas.201162998

[52] H. Hofmann, Statlog (German Credit Data), 1994, (UCI Machine Learning Repository). https://doi.org/10.24432/C5NC77

[53] I. Guyon, A. Saffari, G. Dror, G. Cawley, Agnostic learning vs. prior knowledge challenge, in: 2007 International Joint Conference on Neural Networks, 2007, pp. 829–834. https://doi.org/10.1109/IJCNN.2007.4371065

[54] I. Guyon, S. Gunn, A. Ben-Hur, G. Dror, Gisette, 2008, (UCI Machine Learning Repository). https://doi.org/10.24432/C5HP5B

[55] V. Sigillito, S. Wing, L. Hutton, K. Baker, Ionosphere, 1989, (UCI Machine Learning Repository). https://doi.org/10.24432/C5W01B

[56] W. Wolberg, Breast Cancer Wisconsin (Original), 1992, (UCI Machine Learning Repository). https://doi.org/10.24432/C5HP4Z

[57] M. Kelly, R. Longjohn, K. Nottingham, The UCI machine learning repository, 2024, https://archive.ics.uci.edu. Accessed: 2024-12-20.

[58] Kaggle, Kaggle: High-quality public datasets, 2024. Accessed: 2024-12-20, https://www.kaggle.com/.

[59] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 1–27. https://doi.org/10.1145/1961189.1961199

[60] J.A. Hanley, B.J. McNeil, The meaning and use of the area under a receiver operating characteristic (ROC) curve, Radiology 143 (1) (1982) 29–36. https://doi.org/10.1148/radiology.143.1.7063747

[61] C.J. Van Rijsbergen, Information Retrieval, Butterworth-Heinemann, USA, 2nd edition, 1979.

[62] N. Chinchor, MUC-4 evaluation metrics, in: Fourth Message Understanding Conference (MUC-4): Proceedings of a Conference Held in McLean, Virginia, June 16–18, 1992, 1992.

[63] J. Cohen, A coefficient of agreement for nominal scales, Educ. Psychol. Meas. 20 (1) (1960) 37–46. https://doi.org/10.1177/001316446002000104

[64] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'95, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995, pp. 1137–1143.

[65] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, T. Stützle, M. Birattari, The irace package: iterated racing for automatic algorithm configuration, Oper. Res. Perspect. 3 (2016) 43–58. https://doi.org/10.1016/j.orp.2016.09.002

[66] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30. https://doi.org/10.5555/1248547.1248548

[67] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Annals Mathem. Stat. 11 (1) (1940) 86–92.

[68] A. Benavoli, G. Corani, F. Mangili, Should we really use post-hoc tests based on mean-ranks?, J. Mach. Learn. Res. 17 (5) (2016) 1–10. https://doi.org/10.48550/arXiv.1505.02288

[69] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bulletin 1 (6) (1945) 80–83. https://doi.org/10.2307/3001968

[70] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. 6 (2) (1979) 65–70.

[71] S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, J. Mach. Learn. Res. 9 (89) (2008) 2677–2694.

[72] P.R. Cohen, A.E. Howe, How evaluation guides AI research: the message still counts more than the medium, AI Magazine 9 (4) (1988) 35. https://doi.org/10.1609/aimag.v9i4.952

[73] H. Kim, K.-H. Kim, Deep learning-based SBOM defect detection for medical devices, in: 2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2024, pp. 047–051. https://doi.org/10.1109/ICAIIC60209.2024.10463483

[74] R. Mohammed, J. Rawashdeh, M. Abdullah, Machine learning with oversampling and undersampling techniques: overview study and experimental results, in: 2020 11th International Conference on Information and Communication Systems (ICICS), 2020, pp. 243–248. https://doi.org/10.1109/ICICS49469.2020.239556

[75] J.O. Awoyemi, A.O. Adetunmbi, S.A. Oluwadare, Credit card fraud detection using machine learning techniques: a comparative analysis, in: 2017 International Conference on Computing Networking and Informatics (ICCNI), 2017, pp. 1–9. https://doi.org/10.1109/ICCNI.2017.8123782

[76] M. Azhari, A. Abarda, B. Ettaki, J. Zerouaoui, M. Dakkon, Higgs boson discovery using machine learning methods with pyspark, Procedia. Comput. Sci. 170 (2020) 1141–1146. https://doi.org/10.1016/j.procs.2020.03.053

[77] P. Baldi, P. Sadowski, D. Whiteson, Searching for exotic particles in high-energy physics with deep learning, Nat. Commun. 5 (1) (2014) 4308. https://doi.org/10.1038/ncomms5308

[78] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J.a.P. Fernandes, J.a. Saraiva, Energy efficiency across programming languages: how do energy, time, and memory relate?, in: Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering, SLE 2017, Association for Computing Machinery, New York, NY, USA, 2017, pp. 256–267. https://doi.org/10.1145/3136014.3136031

[79] A.R. Feinstein, D.V. Cicchetti, High agreement but low kappa: i. the problems of two paradoxes, J. Clin. Epidemiol. 43 (6) (1990) 543–549. https://doi.org/10.1016/0895-4356(90)90158-L

[80] M. Hossin, M.N. Sulaiman, A review on evaluation metrics for data classification evaluations, Intern. J. Data Min. Know. Manage. Process 5 (2015) 01–11. https://doi.org/10.5121/ijdkp.2015.5201

[81] Q. Zhang, H. Li, Moea/d: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731. https://doi.org/10.1109/TEVC.2007.892759

[82] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601. https://doi.org/10.1109/TEVC.2013.2281535