Departamento de Ingeniería Informática
y Ciencias de la Computación
Universidad de Concepción

Multiobjective metaheuristics for the support vector machine with feature selection

By
Mathias Felipe Badilla Salamanca

Thesis to obtain the degree of Master of Science in Computer Science

Thesis Advisors
Carlos Emilio Contreras Bolton
Rosa Daniela Medina Durán

March 2026
Concepción Chile

1. Introduction

Supervised classification is an area of growing importance, increasingly applied in everyday contexts. In general terms, this approach relies on datasets that, when subjected to mainly statistical assumptions, enable the construction of models capable of understanding and anticipating the behavior of new samples. This makes it possible to identify patterns and trends in existing data, thereby facilitating informed decision-making and predicting outcomes in previously unobserved scenarios.

Within supervised learning, several classical algorithms are widely employed to address classification tasks, such as decision trees (Quinlan, 1986), k-nearest neighbors (Cover & Hart, 1967), neural networks (Rumelhart et al., 1986), and support vector machines (Cortes & Vapnik, 1995). Among them, the SVM stands out, as it aims to find an optimal hyperplane that separates the classes in the feature space. However, training these algorithms often requires large volumes of data, which entails high computational costs. Moreover, not all available variables contribute valuable information to the model, thus creating the need for: i) algorithms that can be trained efficiently in terms of time and resources, and ii) techniques that allow for the selection of only the most relevant features without sacrificing predictive quality. This latter approach, known as feature selection (FS) (Miao & Niu, 2016), has proven essential for building more accurate and less costly models.

As classification problems have grown in scale and dimensionality, the operations research community has contributed rigorous optimization frameworks to address FS more systematically. Early contributions approached the problem through exact mathematical programming formulations, such as those proposed by Maldonado et al. (2014), Labbé et al. (2019) and Benítez-Peña et al. (2019), which provided provably optimal feature subsets at the cost of high computational demands. However, the inherent complexity of these formulations motivated a progressive shift toward heuristic and decomposition strategies (Aytug, 2015; Gaudioso et al., 2017), which trade optimality guarantees for scalability and computational tractability. Collectively, these contributions have established a solid methodological foundation upon which more sophisticated multi-objective approaches have subsequently been developed.

In the field of operations research, significant progress has been made in the use of multi-objective metaheuristics, applied to classification and FS problems. These methodologies aim to identify solutions that reduce computational resource consumption, particularly training time. Thus, recently, Alcaraz et al. (2022), Valero-Carreras et al. (2023) and Alcaraz (2024) have used a multi-objective metaheuristic approach to address feature selection with suppontvector machines (FS-SVM), and subsequently, Badilla-Salamanca et al. (2025) proposed a new representation of the individual, achieving reduced iteration times and thereby improving the algorithm quality. These results demonstrate that modifying the metaheuristic structure and the individual representation can contribute to reducing iteration time while simultaneously enhancing the predictive model's performance.

These advances are particularly relevant considering that modern models are becoming increasingly complex and expensive, not only in terms of processing time but also due to the economic costs associated with computation (servers, energy, licenses, among others). In scenarios involving large volumes of data, such costs may become a significant barrier. Furthermore, existing multi-objective approaches based on the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Alcaraz et al., 2022; Valero-Carreras et al., 2023; Alcaraz, 2024; Badilla-Salamanca et al., 2025) have primarily focused on structural and empirical risk as optimization objectives, without explicitly incorporating classification performance metrics into the search process. This limitation becomes particularly evident in large-scale and imbalanced datasets, where optimizing solely for margin-based criteria may not adequately reflect the true predictive quality of the resulting models. Optimizing models

for both efficiency and accuracy not only helps reduce these expenses but also increases their applicability in sensitive sectors such as medicine, finance, banking, and logistics, where speed and accuracy of predictions are critical.

In this context, the present work introduces two multi-objective metaheuristic algorithms —NSGA-III, an evolutionary algorithm based on reference-point diversity preservation, and MOPSO, a swarm intelligence method guided by Pareto dominance — integrated with the soft-margin SVM model, a supervised classification framework that tolerates misclassifications through slack variables, for binary classification with feature selection. Both approaches adopt the individual representation proposed by Badilla-Salamanca et al. (2025), incorporating tailored genetic and swarm operators specifically designed to accommodate this encoding. Furthermore, both algorithms simultaneously optimize five objectives encompassing structural risk, empirical risk, and three confusion matrix-based performance metrics, with the dual aim of enhancing predictive quality while reducing computational overhead. Extensive experiments on large-scale benchmark datasets demonstrate that the proposed approaches achieve competitive performance against state-of-the-art algorithms in terms of both predictive metrics and computational efficiency.

1.1. Hypothesis

It is possible to improve the performance, in terms of evaluation metrics and computational time, of a multi-objective metaheuristic based on a soft-margin SVM model with FS, by considering the objectives of empirical risk, structural risk, and the performance of an evaluation metric on large datasets.

1.2. Objectives

Implement an NSGA-III and MOPSO based on the soft-margin SVM model with FS for binary classification datasets.

For this general objective, the following specific objectives are proposed.

- Conduct a literature review.
- Design and implement a NSGA-III to solve large binary classification datasets.
- Design and implement a MOPSO to solve large binary classification datasets.
- Extend the multi-objective SVM-FS formulation by incorporating additional confusion matrix-based performance metrics as explicit optimization objectives.
- Analyze the obtained results and compare them with those reported in the literature in terms of computational time and prediction quality.

1.3. Document outline

The remainder of this document is organized as follows. Section 2 presents a comprehensive literature review covering mathematical programming approaches and population-based metaheuristics for the SVM-FS problem. Section 3 formally introduces the soft-margin SVM model and its extension to the multi-objective SVM-FS framework, detailing the mathematical formulation adopted in this work. Section 4 describes the proposed NSGA-III algorithm, including its solution representation, genetic operators, and overall structure. Section 5 presents the MOPSO algorithm, detailing its particle encoding, velocity update mechanism, and archive management strategy. Section 6 reports the computational experiments conducted on large-scale benchmark datasets, including a description of the datasets, performance metrics, experimental setup, and a statistical analysis of the results. Finally, Section 7 summarizes the main findings of this work and outlines directions for future research.

2. Literature Review

The literature review focuses on examining methodologies that integrate FS mechanisms within SVM architectures. Particular attention is devoted to approaches rooted in the soft-margin SVM paradigm, a formulation that accommodates misclassification tolerance and thus proves suitable for datasets exhibiting non-linear separability (Cortes & Vapnik, 1995; Bishop, 2007). The landscape of existing work spans a diverse spectrum: from rigorous mathematical optimization models to population-based metaheuristics.

The integration of feature selection within SVM learning has been extensively studied through mathematical programming approaches. Maldonado et al. (2014) established a foundational framework by formalizing the combined SVM-FS problem as NP-hard and proposing mixed-integer linear programming (MILP) formulations for its joint optimization. Building on this work, Labbé et al. (2019) introduced budgetary restrictions and tighter hyperplane coefficient bounds to produce more compact classifiers. Decomposition strategies have also been explored: Aytug (2015) applied generalized Benders decomposition to partition the problem into feature selection and SVM training layers, while Gaudioso et al. (2017) employed Lagrangian relaxation to balance model simplicity and accuracy through penalty terms. A pragmatic dimension was introduced by Benítez-Peña et al. (2019), who addressed varying feature acquisition costs via a two-phase MILP and mixed-integer convex quadratic programming approach, a direction extended by Lee et al. (2020) through robust optimization to handle uncertainty in cost estimates. Most recently, Baldomero-Naranjo et al. (2021) broadened the scope by simultaneously addressing outlier detection and FS using a ramp loss-based MILP formulation, demonstrating that heuristic variants can closely approximate optimal solutions at significantly reduced computational cost.

In parallel to the development of exact methods, a distinct research stream has emerged centered on population-based metaheuristics for multi-objective SVM-FS. Unlike single-objective formulations that aggregate competing criteria through weighted sums or penalty parameters, multi-objective methods explicitly treat feature subset size and classification performance as separate, often conflicting objectives. This paradigm shift enables the generation of a Pareto frontier comprising solutions that represent diverse trade-offs between model complexity and predictive accuracy.

The application of the NSGA-II to SVM-FS was pioneered by Alcaraz et al. (2022), who contrasted its performance against the augmented ϵ-constraint 2 (AUGMECON2) technique proposed by Mavrotas & Florios (2013). AUGMECON2 represents an exact multi-objective solver that employs lexicographic optimization, systematically exploring the objective space by parametrically varying constraint bounds. The comparative analysis assessed both the computational efficiency and the quality of Pareto approximations produced by each method within the SVM-FS context.

Extending this line of research, Valero-Carreras et al. (2023) advanced the multi-objective SVM-FS framework by proposing an alternative SVM model that explicitly incorporates confusion matrix-based performance metrics — namely, the AUC-ROC, Cohen's Kappa coefficient, and the F-Score — as evaluation criteria to compare the resulting classifiers against those produced by the classical soft-margin formulation. Their work demonstrated that the choice of performance metrics has a significant impact on the quality of the resulting Pareto approximations, strengthening the practical relevance of multi-objective metaheuristics as a principled alternative to exact methods for the SVM-FS problem.

Further advancements were introduced by Alcaraz (2024), who proposed a comprehensive redesign of the NSGA-II framework tailored to SVM-FS. Key innovations included a novel solution encoding scheme and specialized procedures for eliminating unpromising regions of the search space. These enhancements yielded marked improvements in both convergence rate and overall algorithmic efficiency, surpassing the performance of earlier NSGA-II variants across multiple benchmarks.

Most recently, Badilla-Salamanca et al. (2025) contributed a modified representation strategy for individuals within the NSGA-II population. While retaining the two-objective optimization structure, their approach embeds feature weights directly into the solution encoding, thereby reducing the computational overhead associated with evaluating candidate solutions. This representational shift, coupled with adapted genetic operators, resulted in faster iteration times and improved predictive performance. Furthermore, their work extended the experimental scope of previous studies by evaluating the proposed algorithm on large-scale benchmark datasets, providing insights into its behavior in high-dimensional and computationally demanding scenarios.

Beyond the methodologies explicitly targeting FS within the SVM formulation, complementary research directions have explored related challenges. Some works have focused on the simultaneous optimization of SVM hyperparameters and feature subsets, such as Bouraoui et al. (2018) and Faris et al. (2018), who employed multi-objective genetic and nature-inspired metaheuristics to jointly tune kernel parameters and identify relevant features, enhancing both classification accuracy and model compactness. In parallel, Candelieri et al. (2019) addressed computational scalability through parallel global optimization for hyperparameter tuning, while Sabzekar & Aydin (2021) proposed a noise-aware FS methodology based on relaxed SVM formulations, preserving predictive accuracy under label noise and measurement errors.

The challenge of large-scale, imbalanced datasets was tackled by Dudzik et al. (2021), who developed an evolutionary optimization framework specifically designed to handle class distribution skew. Their method exhibited enhanced robustness and generalization capability in settings where traditional approaches often struggle. Further contributions to the hybrid optimization landscape include the works of Huang et al. (2025), Abasabadi et al. (2022), Jain & Jain (2022), Wang et al. (2023), Bomze et al. (2025), Zandvakili et al. (2024), and Afreen et al. (2025), each introducing novel algorithmic hybridizations and problem-specific adaptations that continue to advance the state of the art.

The most recent wave of innovation in FS has been driven by deep learning architectures, which offer powerful mechanisms for automatic feature discovery in high-dimensional spaces. Unlike the SVM-centric approaches discussed previously, deep learning-based FS methods leverage the representational capacity of neural networks to implicitly or explicitly identify informative features while modeling complex, nonlinear relationships. Applications span diverse domains including bioinformatics, computer vision, and natural language processing.

Several distinct strategies have emerged within this paradigm, including attention mechanisms (Xue et al., 2023; Xue & Zhang, 2024), sparsity-inducing regularization (Farokhmanesh & Sadeghi, 2021), autoencoder-based dimensionality reduction (Bote-Curiel et al., 2022), convolutional transfer learning (Junaid et al., 2025), and reinforcement learning-driven selection (Bouchlaghem et al., 2024). While these approaches offer flexibility and strong empirical performance, they also present notable limitations: training deep networks demands substantial computational resources, the resulting models are often difficult to interpret, and their performance may degrade in scenarios involving small sample sizes or severe class imbalance, potentially limiting their practical utility compared to more transparent approaches such as SVM-based methods.

The literature review indicates that, while exact methods have dominated the SVM-FS landscape, metaheuristic approaches remain comparatively underexplored, particularly in many-objective settings and large-scale scenarios. Building upon the individual representation proposed by Badilla-Salamanca et al. (2025), the present work extends the multi-objective SVM-FS framework. The main contributions of this study are as follows:

- Three classification performance metrics — F1-Score, AUC-ROC, and Cohen's Kappa Coefficient — are incorporated as explicit optimization objectives alongside structural and empirical

risk, resulting in a five-objective formulation that aligns the search process more directly with real-world predictive quality.

- Two population-based algorithms are proposed — an NSGA-III and a MOPSO — with genetic and swarm operators specifically adapted to the solution representation adopted from Badilla-Salamanca et al. (2025), enabling efficient exploration of the expanded objective space while preserving solution feasibility.
- Both algorithms are evaluated on large-scale benchmark datasets where previous approaches have exhibited limited performance, providing empirical evidence of their scalability and competitiveness against state-of-the-art methods in terms of both predictive quality and computational efficiency.
- A rigorous statistical validation framework is employed, combining the Friedman and Iman-Davenport tests for global comparisons, pairwise Wilcoxon signed-rank tests with Bonferroni correction, and the Vargha-Delaney A_{12} effect size statistic, providing a comprehensive and robust assessment of performance differences among the proposed algorithms.

Table 1 summarizes the works reviewed in this section, focusing specifically on multi-objective metaheuristic approaches to the soft-margin SVM-FS problem. For each contribution, the table details the solution method employed, the treatment of the number of selected features — whether fixed as a parameter or determined dynamically — and the number of optimization objectives considered. The table highlights the progressive evolution from exact methods toward population-based metaheuristics, and positions the present work within this landscape by introducing two new algorithms that extend the number of objectives to five while retaining the variable feature selection scheme proposed by Badilla-Salamanca et al. (2025).

Table 1: Summary of the literature review.

Work	**Solution method**	**Number of features**	**Objectives**
Alcaraz et al. (2022)	AUGMECON2 and NSGA-II	Parameter (fixed)	2
Valero-Carreras et al. (2023)	NSGA-II	Parameter (fixed)	2
Alcaraz (2024)	NSGA-II	Parameter (fixed)	2
Badilla-Salamanca et al. (2025)	NSGA-II	Variable	2
This work	NSGA-III MOPSO	Variable	5

3. Support vector machine with feature selection

SVM, introduced by Cortes & Vapnik (1995) and Vapnik (1995), constitute a supervised learning framework for binary classification that seeks an optimal separating hyperplane in an n-dimensional feature space by maximizing the margin between classes. The theoretical foundations of this approach trace back to Vapnik & Chervonenkis (1964), who formalized the construction of optimal separating hyperplanes for pattern recognition, with early applications restricted to linearly separable data (Vapnik & Chervonenkis, 1974). To handle real-world datasets where perfect separation is unattainable, Vapnik (1995) extended the formulation through the soft-margin SVM, incorporating slack variables that permit controlled misclassification. Under this framework, the decision boundary is defined by a subset of training instances known as support vectors, and the learning process simultaneously balances structural risk, associated with margin width, and empirical risk, associated with classification errors.

In this setting, the incorporation of FS becomes particularly relevant. As the dimensionality of the feature space grows, not all variables contribute meaningful information; redundant, noisy, or irrelevant features can deteriorate predictive performance, hinder generalization, and substantially increase computational demands. FS addresses this issue by identifying a compact and informative subset of variables, thereby reducing both model complexity and resource consumption without sacrificing classification quality.

The concept of structural error in SVMs is associated with the separation margin between the two decision hyperplanes. A larger margin implies a more confident boundary, thereby reducing model uncertainty and improving its capacity to generalize to unseen data. Conversely, the empirical error quantifies the mismatch between the predicted and actual labels within the training dataset.

The growing prevalence of SVM-based approaches in supervised learning has brought to light significant scalability challenges, as both training and evaluation become increasingly demanding with larger volumes of instances and higher-dimensional feature spaces.

Several methodological extensions have been proposed to address these limitations. Among them, the relaxed-constraint SVM (Sabzekar et al., 2012) enhances robustness by incorporating mechanisms that better tolerate noise and uncertainty in the data. A complementary strategy is FS, which mitigates the curse of dimensionality by retaining only the most discriminative variables, thereby reducing computational overhead while improving interpretability and generalization.

In the literature, FS techniques are broadly categorized into three paradigms (Guyon et al., 2006; Li et al., 2017). Filter methods assess feature relevance independently of the learning model. Wrapper methods rely on a predictive model to evaluate the contribution of feature subsets to classification performance. Embedded methods, by contrast, integrate feature selection directly into the training process, allowing simultaneous optimization of model parameters and feature relevance.

The present work adopts the embedded paradigm, as feature selection is incorporated into the learning process through the solution encoding of the proposed algorithms.

The soft-margin SVM, introduced by Cortes & Vapnik (1995) and Vapnik (1995), constitutes the foundation of the optimization model adopted in this work. Let $N = \{1, \ldots, n\}$ denote the set of training samples, each characterized by a pair $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, where x_i represents the feature vector and y_i its class label. The dimension of the feature space is d, and the index set of features is defined as $D = \{1, \ldots, d\}$. The decision variables are: w_j $(j \in D)$, the components of the separating hyperplane; b, the bias term; and ξ_i $(i \in N)$, nonnegative slack variables measuring misclassification.

Building upon this formulation, Alcaraz et al. (2022) proposed a multi-objective MICQP model for SVM-FS that decomposes the original objective into two distinct goals — minimizing structural risk f_1 and empirical risk f_2 — while introducing binary variables t_j to control feature selection,

eliminating the dependency on the regularization parameter C:

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

$$\text{minimize } f_2 = \sum_{i \in N} \xi_i \tag{2}$$

subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i \in N \tag{3}$$

$$\xi_i \in \mathbb{R}_0^+ \quad \forall i \in N \tag{4}$$

$$w_j \in \mathbb{R} \quad \forall j \in D \tag{5}$$

$$b \in \mathbb{R} \tag{6}$$

$$\sum_{j \in D} t_j = p \tag{7}$$

$$|w_j| \leq M t_j \quad \forall j \in D \tag{8}$$

$$t_j \in \{0, 1\} \quad \forall j \in D \tag{9}$$

where constraint (7) enforces a fixed number p of selected features, constraint (8) links each feature weight to its selection variable, and constraint (9) defines the binary domain of t_j.

Badilla-Salamanca et al. (2025) extend the multi-objective MICQP formulation proposed by Alcaraz et al. (2022), introducing two key modifications to the model. First, they reformulate the objective function f_2 in (2) by replacing it with the new expression shown in (10). This modified objective incorporates the parameter α_i ($i \in N$), which weights the classification errors according to the class distribution of the dataset.

Second, the original constraint (7), which imposed a fixed number of selected features p, is replaced by a dynamic constraint that retains the same left-hand side but introduces an adaptive upper bound based on the total number of available features d, as defined in (11). This modification enables the algorithm to dynamically adjust the number of selected features and converge toward an optimal subset, in contrast to the approach proposed by Valero-Carreras et al. (2023), who fixed p to five features.

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

$$\sum_{j \in D} t_j \leq d \tag{11}$$

Consequently, the model proposed by Badilla-Salamanca et al. (2025) incorporates SVM constraints and multiobjective objetive function and constraints 8, 9 from Alcaraz et al. (2022). This formulation allows for the selection of any number of features—from a single variable to the entire set.

$$\text{minimize } f_1 = \frac{1}{2}|w|^2 \tag{12}$$

$$\text{minimize } f_2 = \sum_{i \in N} \alpha_i \xi_i \tag{13}$$

subject to:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i \in N \tag{14}$$
$$\xi_i \in \mathbb{R}_0^+ \quad \forall i \in N \tag{15}$$
$$w_j \in \mathbb{R} \quad \forall j \in D \tag{16}$$
$$b \in \mathbb{R} \tag{17}$$
$$|w_j| \leq M t_j \quad \forall j \in D \tag{18}$$
$$t_j \in \{0, 1\} \quad \forall j \in D \tag{19}$$
$$\sum_{j \in D} t_j \leq d \tag{20}$$

This work extends the model proposed by Badilla-Salamanca et al. (2025) by introducing three additional optimization objectives oriented toward classification performance: the *F1-score*, the *AUC-ROC*, and the *Cohen's kappa coefficient*, all of which are metrics derived from the confusion matrix. Accordingly, the final model to be optimized is defined as follows:

$$\text{minimize } f_1 = \frac{1}{2}|w|^2 \tag{21}$$
$$\text{minimize } f_2 = \sum_{i \in N} \alpha_i \xi_i \tag{22}$$
$$\text{maximize } F1 \tag{23}$$
$$\text{maximize } AUC - ROC \tag{24}$$
$$\text{maximize } CK \tag{25}$$
subject to:
$$y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i \in N \tag{26}$$
$$\xi_i \in \mathbb{R}_0^+ \quad \forall i \in N \tag{27}$$
$$w_j \in \mathbb{R} \quad \forall j \in D \tag{28}$$
$$|w_j| \leq M t_j \quad \forall j \in D \tag{29}$$
$$t_j \in \{0, 1\} \quad \forall j \in D \tag{30}$$
$$\sum_{j \in D} t_j \leq d \tag{31}$$
$$b \in \mathbb{R} \tag{32}$$

4. NSGA-III

Metaheuristics are high-level optimization strategies designed to efficiently explore large and complex search spaces where exact methods become computationally intractable. Among the most widely adopted frameworks for multi-objective optimization is the NSGA-II (Deb et al., 2002) and NSGA-III (Deb & Jain, 2014). NSGA-III combines fast non-dominated sorting with a crowding distance mechanism to maintain a diverse and well-distributed approximation of the Pareto front. While NSGA-II has proven effective for problems with two or three objectives, its crowding distance operator becomes unreliable as the number of objectives grows, leading to poor diversity preservation in high-dimensional objective spaces.

NSGA-III is an evolutionary algorithm for multi-objective problems with "many" objectives (≥ 4) that preserves the general structure of NSGA-II (initial population, genetic operators, and non-dominated sorting), but replaces the diversity operator with a scheme guided by reference points that are regularly distributed on a normalized hyperplane (Deb & Jain, 2014). The central idea is to emphasize non-dominated solutions that are associated with those references, which improves coverage of the Pareto front in high-dimensional spaces.

This NSGA-III proposal is based on the NSGA-II implementation proposed by Badilla-Salamanca et al. (2025), and extends it to a many-objective setting by incorporating additional classification performance metrics as explicit optimization objectives. This extension is motivated by the need to simultaneously optimize multiple dimensions of classifier performance without sacrificing one metric in favor of another, enabling the algorithm to approximate a Pareto front that explicitly captures the trade-offs among competing goals and yields solutions that are well-balanced across all desired dimensions of predictive quality.

4.1. Overall algorithm

Algorithm 1 describes the operation of NSGA-III, a multi-objective evolutionary algorithm specifically designed for problems with high-dimensional objective spaces, where traditional approaches may suffer from loss of diversity or an excessive concentration of solutions in limited regions of the Pareto front. The algorithm preserves the core evolutionary structure of NSGA-II — including non-dominated sorting, genetic operators, and elitist population update — while introducing an enhanced diversity-preservation mechanism based on reference points, enabling superior performance in high-dimensional multi-objective optimization problems.

The process begins by generating an initial population of size $\mathcal{K}$ using INITIAL-POPULATION (line 1). Each individual is evaluated through the function EVALUATE-POPULATION (line 2). A fast non-dominated sorting procedure—the same mechanism employed in NSGA-II—is then applied to classify solutions into fronts $F_1, F_2, \ldots$, where the best non-dominated solutions are placed in F_1 (line 4).

Reference points are generated internally using GENERATE-REFERENCE-POINTS (line 3) with $M = 5$ objectives and $H = 12$ divisions, which determines the granularity of the reference point distribution on a normalized hyperplane, as further detailed in Section 4.2.

In each evolutionary iteration, a temporary offspring population Q of size $\mathcal{K}/2$ is created (line 6) by pairing consecutive individuals from the population (PARENT-SELECTION), as described later. After applying genetic operators (lines 9–13), the offspring are evaluated (line 15) and merged with the parent population to form $R = P \cup Q$ (line 16), which is sorted again using FAST-NON-DOMINATED-SORT (line 17).

To construct the next population, NSGA-III initially follows the same strategy as NSGA-II: it incorporates all complete fronts $F_1, \ldots, F_m$ that fit entirely within the population size $\mathcal{K}$ (lines 19–20).

The key distinction with respect to NSGA-II arises when the next front F_{m+1} cannot be included *completely* within the population size constraint. While NSGA-II resolves this situation using crowding distance, NSGA-III replaces this mechanism with a reference-point-based niching procedure implemented through SELECTION-REFPOINT-NICHING (line 21), which maintains an effective distribution of solutions in many-objective scenarios where crowding distance becomes unreliable, yielding more stable and better-distributed approximations to complex Pareto fronts.

The evolutionary loop continues until the time limit T is reached, at which point the final population P is returned as the approximation to the Pareto front obtained by NSGA-III.

Algorithm 1: NSGA-III

Input: $\mathcal{K}$, T, M,H, ρ^1, ρ^2
Output: P

```
1  P ← initial-population (K)
2  evaluate-population (P)
3  Z ← generate-reference-points (M, H)
4  F ← fast-non-dominated-sort (P)
5  while a given time limit T is not exceeded do
6      Q ← {∅}
7      for k ← 1 to K/2 do
8          (a, e) ← parent-selection (P)
9          i ← crossover (a, e)
10         if random(0, 1) ≤ ρ^1 then
11             i ← mutation-1(i)
12         if random(0, 1) ≤ ρ^2 then
13             i ← mutation-2(i)
14         Q ← Q ∪ {i}
15     evaluate-population (Q)
16     R ← P ∪ Q
17     F ← fast-non-dominated-sort (R)
18     P ← ∅
19     m ← get-complete-front-limit(F, K)
20     P ← merge-complete-fronts(F, m)
21     P ← P ∪ selection-refpoint-niching(F_{m+1}, Z, K − |P|)
```

4.2. Reference-point niching procedure

The `selection-refpoint-niching` procedure is invoked when the next front F_{m+1} cannot be fully incorporated into the new population. It operates on the normalized objective space and consists of four sequential steps that collectively ensure a balanced coverage of the Pareto front, leveraging the reference points Z generated by `generate-reference-points`, which receives as parameters M, the number of objectives, and H, the number of divisions on the normalized hyperplane that controls the granularity of the reference point distribution. In this work, $M = 5$ and $H = 12$ are used, yielding a structured set of reference points that adequately covers the five-dimensional objective space.

1. **Normalization:** The objective vectors of all solutions in fronts F_1 through F_{m+1} are normalized using the ideal point (minimum value for each objective) and the extreme points (solutions that maximize individual objectives) to map all objectives to the range $[0, 1]$. This

ensures that all objectives contribute proportionally to the distance calculations, regardless of their original scales.

2. **Association:** Each solution in F_{m+1} is associated with its closest reference point based on perpendicular distance in the normalized objective space. Specifically, for each solution $\mathbf{s}$, the algorithm computes the perpendicular distance from $\mathbf{s}$ to each reference line (the line connecting the origin to each reference point), and assigns $\mathbf{s}$ to the reference point that minimizes this perpendicular distance.
3. **Niche identification:** The algorithm identifies underrepresented niches by counting how many solutions from the current population $(F_1, \ldots, F_m)$ are already associated with each reference point. Reference points with fewer associated solutions represent underexplored regions of the objective space.
4. **Selection:** Solutions are selected from F_{m+1} to fill the remaining population slots, prioritizing those associated with the least populated reference points. If a reference point has no associated solutions in the current population, the solution from F_{m+1} closest to that reference point is selected first. This ensures balanced coverage of the objective space and prevents clustering in any particular region.

.

4.3. Solution representation

Each solution within the population P is represented by three chromosomes (π^1, π^2, π^3). The first chromosome, π^1, encodes the identifiers of the selected features. The second chromosome, π^2, stores the real-valued weights assigned to the selected features, i.e., $\pi^2 \in \mathbb{R}$, and has the same dimensionality as π^1. These weights determine the relative contribution of each feature to the classification decision. The third chromosome, π^3, contains a single scalar value corresponding to the model's intercept term (b). Figure 1 illustrates the encoding of an individual with p selected features, where π^1 represents the indices of the active features, π^2 their corresponding weights, and π^3 the intercept.

To exemplify this representation, consider $\pi^1 = (1, 2)$, $\pi^2 = (1, -1)$, and $\pi^3 = -1$. In this case, features 1 and 2 are selected with associated weights of 1 and -1 respectively, and an intercept of -1. The resulting decision boundary is expressed as $x_1 - x_2 - 1 = 0$, while the two margin hyperplanes are obtained by shifting this equation to 1 and -1 respectively.

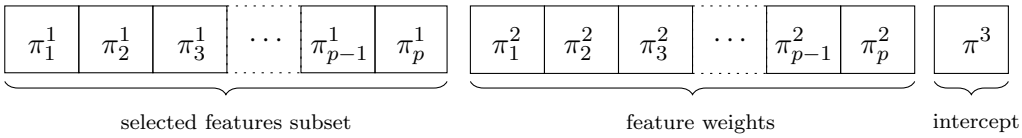

Figure 1: Representation of an individual solution proposed by Badilla-Salamanca et al. (2025)

4.4. Initial population

In the initial population, the three chromosomes of each individual are generated through a randomized process that ensures diversity in the search space. The first chromosome, π^1, is constructed by randomly selecting a subset of features from a predefined interval $[f^l, f^u]$, guaranteeing that all features have the same likelihood of being included in the individual. Once the subset is defined, the second chromosome, π^2, assigns a corresponding real-valued weight to each selected feature, where these weights are drawn uniformly from the range $[w^l, w^u]$. The third chromosome, π^3, represents the intercept term of the model and is initialized with a random value within a predefined interval $[b^l, b^u]$.

4.5. Selection

Parent selection in this implementation pairs consecutive individuals from the population to produce offspring for the next generation. Following the application of `fast-non-dominated-sort`, the population is structured according to non-dominated fronts, with higher-quality solutions belonging to F_1 positioned earlier in the population array, and lower-ranked solutions placed progressively toward the end.

By pairing consecutive individuals — that is, positions $2k-1$ and $2k$ form a pair, for $k = 1, \ldots, \mathcal{K}/2$ — the `parent-selection` strategy implements a form of assortative mating that tends to match solutions of similar quality. This design ensures that solutions from less populated and underexplored regions of the Pareto front, which are prioritized by the reference-point niching mechanism, are more likely to be paired with solutions of comparable rank, thereby promoting balanced and diverse offspring generation across the objective space. Furthermore, this pairing guarantees that all $\mathcal{K}/2$ pairs produce exactly one offspring each, maintaining a consistent population size throughout the evolutionary process.

4.6. Crossover

Crossover serves as a key genetic operator that merges information from two parent solutions to produce new offspring, thereby promoting a broader exploration of the search space. This mechanism enables offspring to inherit traits from both parents, fostering convergence toward a well-distributed set of non-dominated solutions in multi-objective optimization contexts.

In the operator, the first chromosome π^1, which encodes the selected feature identifiers, and the second chromosome π^2, which contains their corresponding weights, are both divided into l equal segments of length L. Given two parent solutions, a and e, generate a new offspring, i, by combining these segments. Specifically, the first L elements of π^1_a are copied into π^1_i, after which the first element in π^1_e greater than the last copied feature is located, and the next L unique elements are added. If π^1_e does not contain enough distinct features, the remaining positions are filled with unused identifiers from π^1_a, ensuring no repetitions. This mechanism may produce offspring with slightly fewer features, justifying the lower bound set for feature selection during initialization.

The operation is illustrated in Figure 2, where $L = 2$ and both parents have chromosomes of length six. The offspring's feature vector is formed by interleaving segments from each parent: the first third from parent a, the second from parent e, and the final part again from a. Note that in this example, the last segment does not perfectly follow $L = 2$ because feature 5, inherited from parent a, was already included in the earlier segment from parent e, leaving only feature 6 to be added at the end. This process ensures that the offspring inherits a non-redundant and ordered combination of features from both parents. The corresponding weights in π^2 are inherited from the same parent that contributed each feature, while the intercept chromosome π^3 is calculated as the mean of the intercepts of the two parents.

4.7. Mutation

Mutation introduces stochastic variation into the population, promoting exploration of new regions in the search space and reducing the risk of premature convergence. By modifying individuals at random, this operator helps maintain genetic diversity and enhances the algorithm's ability to escape local optima.

Two complementary mutation operators are employed. The first modifies only the values of an individual's existing components, while the second changes the number of active features without altering the model's feasibility. Both operators are detailed below.

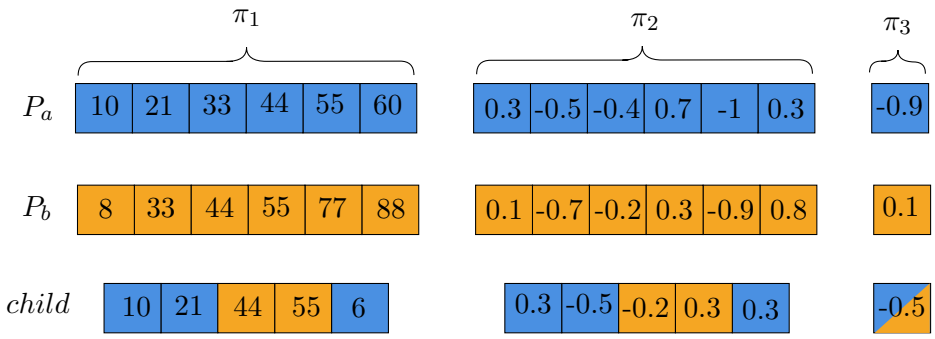

Figure 2: Example of the proposed crossover operator where parent chromosomes a and e generate offspring i. Badilla-Salamanca et al. (2025)

- *mutation-1*: This operator randomly alters one or more attributes of an individual. A selected feature identifier may be replaced by another unselected one; feature weights can be increased or decreased by 5%; and the intercept value may vary by $\pm 10\%$. All non-selected features have equal probability of being chosen, and each of these three operations occurs independently with a probability of 33%. Although the expected number of modifications per mutation is one, multiple changes can occur simultaneously.

- *mutation-2*: This operator varies the number of selected features, increasing or decreasing it with equal likelihood. The number of features to add or remove is determined by a discrete uniform distribution and ranges from one up to 20% of the individual's current feature count. Removal operations are applied only when the individual contains more than seven features, ensuring the feasibility of the resulting solution.

5. MOPSO

Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995) is a population-based metaheuristic inspired by the collective movement of biological swarms, such as flocks of birds or schools of fish. In PSO, a set of candidate solutions, referred to as particles, navigate the search space by updating their positions and velocities at each iteration. Each particle is influenced by two components: a cognitive component, which attracts it toward its own best-known position, and a social component, which guides it toward the best position found by the swarm. This balance between individual experience and collective knowledge enables efficient exploration and exploitation of the search space.

MOPSO (Coello Coello & Lechuga, 2002) extends this framework to handle multi-objective optimization problems by incorporating Pareto dominance to determine the movement direction of the particles, maintaining an external repository of non-dominated solutions discovered throughout the search. This chapter introduces the proposed MOPSO, providing a general definition of the algorithm as well as a detailed description of its operators and their functioning.

5.1. Overall algorithm

Algorithm 2: MOPSO

Output: $\mathcal{R}_X$, $\mathcal{R}_F$

1 $X \leftarrow$ `initial-population` (n_{POP})
2 $V \leftarrow$ `initial-velocity` (n_{POP})
3 $F \leftarrow$ `evaluate` (X)
4 $\mathbf{P}^*_X \leftarrow$ `best_position(`X`)`
5 $\mathbf{P}^*_F \leftarrow$ `best_fitness(`F`)`
6 $\mathcal{R}_X, \mathcal{R}_F \leftarrow$ `non-dominated-particles` (X, F)
7 $\mathcal{G} \leftarrow$ `adaptive-grid` $(\mathcal{R}_F, n_{\text{Div}})$
8 $(g_p, g_q) \leftarrow$ `grid-index` $(\mathcal{R}_F, \mathcal{G})$
9 $s \leftarrow n_{\text{POP}}$
10 **while** a given time limit T is not exceeded **do**
11 $\quad w \leftarrow 0.9 - \dfrac{0.5}{S_{\max}} \cdot s$
12 $\quad$ **for** $i \leftarrow 1$ **to** n_{POP} **do**
13 $\quad\quad h \leftarrow$ `select-leader` $(\mathcal{R}_F, g_p)$
14 $\quad\quad V_i \leftarrow wV_i + r_1c_1(\mathbf{P}^*_{X_i} - X_i) + r_2c_2(\mathcal{R}_{Xh} - X_i)$
15 $\quad\quad X_i \leftarrow X_i + V_i$
16 $\quad\quad X_i \leftarrow$ `apply-feature-threshold` (X_i, θ)
17 $\quad\quad X_i, V_i \leftarrow$ `maintain-bounds` $(X_i, V_i, v_{\max})$
18 $\quad F \leftarrow$ `evaluate` (X)
19 $\quad \mathbf{P}^*_X, \mathbf{P}^*_F \leftarrow$ `update-personal-best` $(X, F, \mathbf{P}^*_X, \mathbf{P}^*_F)$
20 $\quad \mathcal{R}'_X, \mathcal{R}'_F \leftarrow$ `non-dominated-particles` (X, F)
21 $\quad \mathcal{R}_X, \mathcal{R}_F, g_p, g_q \leftarrow$ `archive-controller` $(\mathcal{R}_X, \mathcal{R}_F, \mathcal{R}'_X, \mathcal{R}'_F, g_p, g_q)$
22 $\quad$ **if** *particles outside grid bounds* **then**
23 $\quad\quad \mathcal{G} \leftarrow$ `adaptive-grid` $(\mathcal{R}_F, n_{\text{Div}})$
24 $\quad\quad (g_p, g_q) \leftarrow$ `grid-index` $(\mathcal{R}_F, \mathcal{G})$
25 $\quad$ **if** $|\mathcal{R}_F| > n_{Rep}$ **then**
26 $\quad\quad \mathcal{R}_F, \mathcal{R}_X, g_p, g_q \leftarrow$ `remove-particles` $(\mathcal{R}_F, \mathcal{R}_X, n_{\text{Rep}}, g_p, g_q)$
27 $\quad s \leftarrow s + n_{\text{POP}}$

Algorithm 2 presents the structure of MOPSO. The swarm is initialized with random particle positions X and zero velocities V (lines 1–2). Each particle maintains a personal best position ($\mathbf{P}^*_X$) and fitness ($\mathbf{P}^*_F$), initially set to their starting configuration (lines 4–5). An external repository ($\mathcal{R}_X$, $\mathcal{R}_F$) stores all non-dominated solutions discovered throughout the search (line 6), serving as an elite archive that preserves high-quality solutions. An adaptive grid $\mathcal{G}$ partitions the objective space into n_{Div} hypercubes per dimension (line 7), assigning each repository solution to a cell indexed by g_p and g_q based on its objective values (line 8). This structure enables the algorithm to identify sparsely and densely populated regions of the Pareto front (lines 10–27).

The main loop iterates until time limit T is reached. At each iteration, the inertia weight w decreases linearly from 0.9 to 0.4 as s grows toward $S_{\max}$ (line 11), progressively shifting the search from exploration to exploitation. For each particle i, a guide particle h is selected from $\mathcal{R}_F$ using fitness-based roulette wheel selection (line 13), where hypercubes containing fewer particles receive higher selection probability — specifically, the fitness assigned to a hypercube is $10/n_p$, where n_p is the number of particles it contains. This mechanism promotes selection of guides from underexplored regions.

The velocity update equation (line 14) combines three components: (1) inertia (wV_i), which preserves the particle's current flight direction and decreases over time; (2) cognitive attraction ($r_1c_1(\mathbf{P}^*_{X_i} - X_i)$), pulling the particle toward its personal best experience; and (3) social attraction ($r_2c_2(\mathcal{R}_{Xh} - X_i)$), drawing it toward the selected repository guide. Here, r_1 and r_2 are random values uniformly distributed in $[0, 1]$ that introduce stochasticity, while c_1 and c_2 are acceleration coefficients (set to 2.8 and 1.3, respectively) that control the relative influence of cognitive and social components. After updating positions (line 15), APPLY-FEATURE-THRESHOLD (line 16) deactivates features with selection probability below $\theta = 0.5$ by setting their corresponding weights to zero, ensuring consistency in the SVM-FS solution representation. Velocity and position bounds are then enforced through MAINTAIN-BOUNDS (line 17) to maintain particles within the valid search space, with velocities limited to $v_{\max}$ and directions reversed when boundaries are exceeded.

Following evaluation (line 18), personal memories $\mathbf{P}^*_X$ and $\mathbf{P}^*_F$ are updated using Pareto dominance criteria (line 19): if the current position dominates the stored best, it replaces the memory; if the stored best dominates the current position, the memory is retained; if neither dominates the other, one is selected randomly with equal probability. The repository is then refreshed through ARCHIVE-CONTROLLER (line 21), which merges the newly discovered non-dominated solutions $\mathcal{R}'_X$, $\mathcal{R}'_F$ with the existing repository $\mathcal{R}_X$, $\mathcal{R}_F$ and eliminates any solutions that have become dominated. This operation may also remove duplicate solutions to maintain repository integrity.

The adaptive grid $\mathcal{G}$ is recalculated whenever new solutions fall outside the current objective bounds (line 22). This recalculation updates the minimum and maximum values for each objective, reconstructs the hypercube subdivisions, and reassigns all repository solutions to their new corresponding cells indexed by g_p and g_q (lines 23–24). Although this operation incurs computational cost proportional to the repository size and number of objectives, it occurs selectively — only when bounds change — rather than at every iteration. When the repository exceeds its maximum capacity n_{Rep} (line 25), REMOVE-PARTICLES (line 26) eliminates solutions from the most crowded hypercubes, applying the same fitness-based roulette wheel mechanism but inversely: hypercubes with more particles receive higher elimination probability. This pruning strategy maintains diversity by preventing clustering in any particular region of the objective space.

The algorithm terminates after the time limit is reached, returning the final repository contents $\mathcal{R}_X$ and $\mathcal{R}_F$ as the approximation to the Pareto front.

5.2. Solution representation

For a set of d variables, each particle in the population is defined as a vector of length $2d + 1$, composed of three distinct segments. The first segment (π_1), of size d, encodes the probabilities of considering each variable. Higher values increase the likelihood of a variable being included in the solution, whereas lower values result in the corresponding regressor being deactivated during prediction. The second segment (π_2), also of size d, contains the regressors associated with the variables. The final element (π_3) corresponds to the intercept of the model. All vectors consist of continuous values, with the probabilities constrained to the interval $[0, 1]$. A representation of the individual is illustrated in Figure 3.

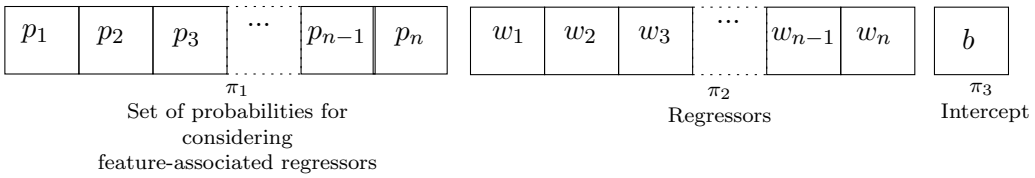

Figure 3: Structure of a particle in the proposed MOPSO, represented as a vector of length $2n + 1$, composed of feature-selection probabilities, regressors, and the intercept.
Badilla-Salamanca et al. (2025)

Continuous encoding allows exploring the solution space with smooth transitions between the inclusion and exclusion of each variable, facilitating a progressive assessment of its contribution to model performance. Simultaneously considering per-feature inclusion scores (with a threshold) and the regressor's parameters enables co-optimization of the number of features and the model fit (Xue et al., 2016).

5.3. Initialization

The initialization process constitutes the first stage of the MOPSO algorithm and aims to generate an initial set of particles that represent a diverse distribution in the search space. Each particle i is characterized by its position X_i, its velocity V_i, and its personal memory $\texttt{PBEST}_i$. In this phase, the initial conditions of both positions and velocities are established, ensuring that the solutions are valid and that the swarm possesses sufficient diversity to effectively explore the solution space.

Position initialization. Each position X_i is generated randomly within the lower and upper bounds defined for each decision variable. Since the position vector is composed of three sections—(i) the probabilities of feature selection, (ii) the weights associated with the activated features, and (iii) a bias term—each component is initialized according to its predefined limits.

During initialization, a logical constraint is applied to ensure coherence between the selection vector and its corresponding weights: if a feature f_j is not activated ($f_j < 0.5$), the associated weight w_j is set to zero.

This guarantees that the solution representation remains consistent with the effective selection of variables and prevents exploration of invalid combinations. This procedure is executed through the `initial-population` function followed by `main-features`, ensuring that the initial swarm is randomly distributed while still respecting the problem constraints.

Velocity initialization. Initial velocities V_i are set to zero for all particles, following the guidelines of Coello Coello & Lechuga (2002):

$$V_i = \mathbf{0}. \tag{33}$$

Additionally, a maximum velocity per variable, $v_{\max}$, is defined to limit particle displacement within a stable range:

$$v_{\max} = 2 \times \frac{u_b - l_b}{n_{\text{steps}}}. \tag{34}$$

This limit acts as a damping factor and prevents numerical instability during the iterations.

Initial evaluation and personal memories. Once the initial positions have been generated, each particle is evaluated using the objective function $\mathcal{F}(X_i)$, which produces an objective vector

$$F_i = [f_1(X_i), f_2(X_i), \ldots, f_5(X_i)]. \tag{35}$$

These initial values are used to define the particles' personal memories:

$$\mathbf{P}^*_{X_i} = X_i, \quad \mathbf{P}^*_{F_i} = F_i, \tag{36}$$

that is, the best-known position for each particle at this stage corresponds to its initial state.

5.4. Diversification of solutions

The preservation of diversity within the set of non-dominated solutions is achieved through an adaptive grid in objective space, following Coello Coello & Lechuga (2002). This mechanism ensures a balanced distribution of solutions maintained in an external repository—a secondary population storing only non-dominated vectors— thereby preventing excessive concentration in specific regions of the Pareto front.

First, after each particle evaluation, the set of non-dominated solutions—represented by their objective vectors—is obtained using the function `non-dominated-particles`. From this set, the algorithm constructs a partitioning structure of the objective space: each dimension of the space is divided into n_{Div} intervals, thus generating a collection of hypercubes or cells that cover the current performance region. The grid adapts dynamically, as the minimum and maximum values of each objective function are updated as the Pareto front evolves. This ensures that the search space accurately reflects the region of interest at each iteration. The adaptation process is carried out through the `adaptative-grid` function.

Each non-dominated solution is assigned to one of the hypercubes according to its objective values using the `grid-index` function. In this way, the algorithm can quantify the density of solutions within each cell, information that is used for two fundamental purposes:

1. **Leader selection:** particles preferentially select their leader from the repository solutions located in less populated hypercubes, thus promoting the exploration of sparsely visited regions of the objective space.
2. **Repository maintenance:** when the repository reaches its maximum capacity, solutions belonging to the most crowded hypercubes are removed with higher probability, fostering the preservation of diversity.

As the Pareto front evolves during execution, the grid is recalculated whenever new solutions fall outside the current objective bounds, requiring that limits be updated, subdivisions reconstructed, and solutions reassigned to new hypercubes. This adaptive mechanism maintains both convergence toward the Pareto front and uniform distribution across objective space, as proposed in Coello Coello & Lechuga (2002).

5.5. Velocity and position updates

The velocity and position of the particles form the fundamental basis of MOPSO, since each particle represents a candidate solution to the optimization problem. Therefore, the update process of these variables is critical, as it governs the exploration and exploitation of the search space and directly influences the convergence toward the Pareto front.

The velocity update of particle i is carried out according to Equation 37, where w is the inertia weight of the particle. This inertia decreases linearly over the iterations, as described in Equation 38, with a decrement of 5×10^{-4} per iteration. $\mathbf{P}^*_{X_i}$ represents the best position that particle i has achieved so far, while X_i denotes its current position.

The parameters r_1 and r_2 are random numbers uniformly distributed in the range $[0, 1]$, introducing stochasticity into the movement of the particles. The term $(\mathbf{P}^*_{X_i} - X_i)$ corresponds to the cognitive component, which attracts each particle toward its personal best position. Meanwhile,

$(\mathcal{R}_{Xh} - X_i)$ represents the social component, which guides the particle toward a leader selected from the repository of non-dominated solutions.

The index h is determined through a fitness-based selection mechanism applied to the adaptive grid $\mathcal{G}$. Hypercubes containing more than one particle are penalized by assigning them a fitness value equal to x/n_p, where $x > 1$ (commonly $x = 10$) and n_p is the number of particles within the hypercube. This mechanism promotes diversity by discouraging overcrowding in certain regions of the search space. Subsequently, a roulette-wheel selection (Coello Coello & Lechuga, 2002) is applied using these fitness values to choose the hypercube, and within it, a leader particle is selected at random.

$$V_i = wV_i + r_1 \cdot c_1 \cdot (\mathbf{P}^*_{X_i} - X_i) + r_2 \cdot c_2 \cdot (\mathcal{R}_{Xh} - X_i) \tag{37}$$

$$w = 0.9 - \frac{0.5}{S_{\max}} \cdot s \tag{38}$$

The position of particle i is adjusted according to Equation 39.

$$X_i = X_i + V_i \tag{39}$$

6. Computational experiments

This section presents the computational experiments conducted to evaluate the performance of the proposed NSGA-III and MOPSO algorithms. The experimental design follows the methodology established by Badilla-Salamanca et al. (2025), ensuring consistency with prior work and enabling a direct comparison against the state-of-the-art NSGA-II algorithm. The section is organized as follows: Section 6.1 describes the large-scale benchmark datasets employed in the evaluation; Section 6.2 defines the performance metrics used for comparison; Section 6.3 details the experimental protocol; and Section 6.5 presents and discusses the obtained results, including a rigorous statistical significance analysis.

6.1. Dataset description

The proposed algorithm is assessed through computational experiments conducted on binary classification datasets spanning a range of dimensionalities and feature counts that are widely used in the literature. For comparative analysis, we benchmark our method against the NSGA-II approach of Badilla-Salamanca et al. (2025), focusing exclusively on the large-scale datasets employed in their study, where their approach exhibited notably limited performance. A concise description of each dataset is presented below.

- **Software** (Kim & Kim, 2024): Derived from industrial software systems and used to predict defect-prone modules.
- **Santander** (Mohammed et al., 2020): A financial application aimed at classifying customer behavior from anonymized transaction records.
- **Fraud** (Awoyemi et al., 2017): A credit-card fraud detection benchmark characterized by extreme class imbalance.
- **Higgs** (Azhari et al., 2020): Particle-physics data for discriminating signal from background events.
- **SUSY** (Baldi et al., 2014): Simulated high-energy physics events; the largest and one of the most challenging datasets in our study.

Figure 4 shows the class distribution for each dataset, highlighting in particular the cases of Santander and Fraud, where the class imbalance is clearly evident. All datasets were obtained from the Kaggle platform (Kaggle, 2024). Table 3 summarizes the sample sizes used in our experiments.[1]

Table 2: Characteristics of datasets.

Dataset	n	d
Software	101,763	20
Santander	200,000	200
Fraud	284,807	30
Higgs	1,000,000	30
SUSY	5,000,000	18

[1]Sample sizes correspond to the versions and preprocessing adopted in this study. For *Higgs*, we used a fixed subsample to align with the experimental budget.

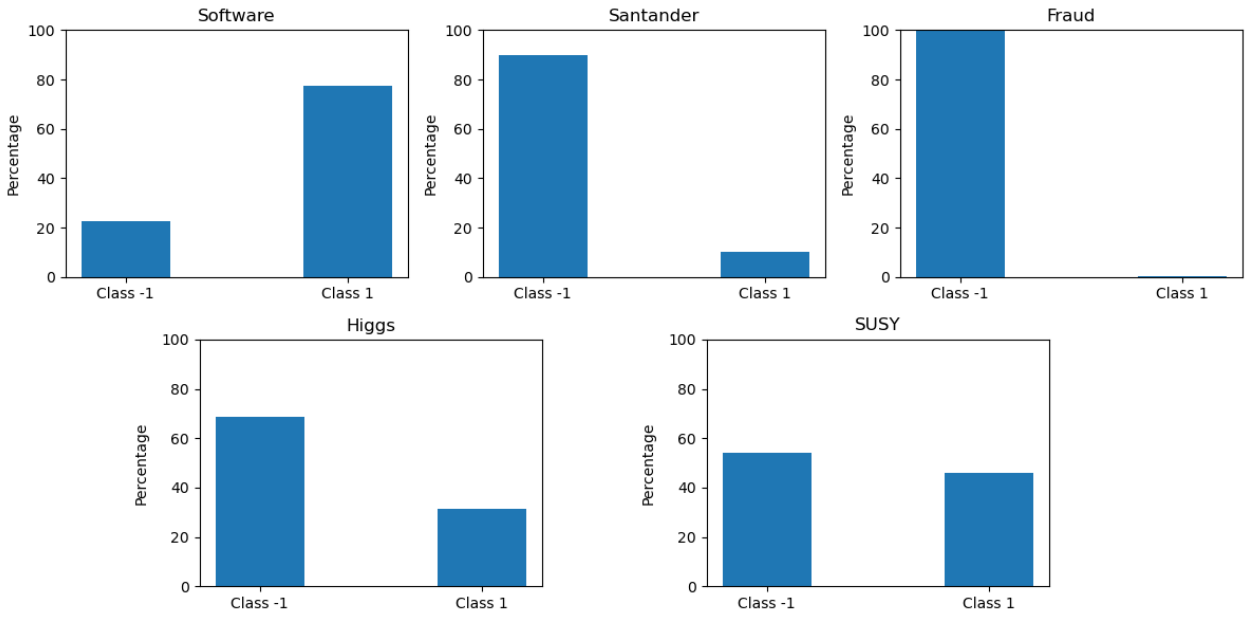

Figure 4: Datasets class distribution

6.2. Performance Metrics

The evaluation of the proposed algorithm relies on the same three performance measures used by Valero-Carreras et al. (2023), ensuring methodological consistency and enabling a fair comparison with previous studies. These metrics are derived from the confusion matrix, a standard analytical tool used to evaluate classification models.

The confusion matrix provides a detailed view of model performance by contrasting predicted and actual class labels (Bishop, 2007). It enumerates true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), enabling an in-depth assessment of prediction outcomes. From this matrix, several metrics are derived—namely, the area under the receiver operating characteristic curve (AUC-ROC) (Hanley & McNeil, 1982), the F-Score (Van Rijsbergen, 1979; Chinchor, 1992), and Cohen's Kappa coefficient (CKC) (Cohen, 1960)—which together capture different aspects of model accuracy, balance, and reliability.

- **AUC-ROC:** This metric evaluates the relationship between the true positive rate (TPR) (Equation (40)) and the false positive rate (FPR) (Equation (41)) over multiple classification thresholds. The AUC quantifies the probability that a randomly chosen positive instance is assigned a higher score than a randomly chosen negative one. Computed using class probability estimates (often derived via a sigmoid function), AUC-ROC values range from 0 to 1, where higher values indicate greater discriminative capacity.

$$\text{TPR} = \frac{\text{TP}}{\text{TP+FN}} \tag{40}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP+TN}} \tag{41}$$

- **F-Score:** This measure combines *Precision* (Equation (42)) and *Recall* (Equation (43)) into a single harmonic mean, balancing both the correctness of positive predictions and the model's sensitivity to all positive cases. The F1-Score is particularly informative under class imbalance

or when misclassification costs differ between classes, taking values between 0 and 1, with higher scores denoting stronger performance.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{42}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{43}$$

$$\text{F-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{44}$$

- **CKC:** This statistic measures the level of agreement between predicted and actual classifications, correcting for the agreement expected by chance. Its range extends from -1 (complete disagreement) to 1 (perfect agreement), with 0 representing random consistency. CKC is especially valuable in evaluating models trained on imbalanced datasets or comparing categorical assessments. It is computed using Equations (45)–(47), where p_o denotes the observed agreement and p_c the expected agreement by chance.

$$p_o = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{45}$$

$$p_c = \frac{(\text{TP} + \text{FP}) \times (\text{TP} + \text{FN}) + (\text{TN} + \text{FN}) \times (\text{TN} + \text{FP})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})^2} \tag{46}$$

$$\text{CKC} = \frac{p_o - p_c}{1 - p_c} \tag{47}$$

6.3. Experiment setup

The experiments were conducted on a personal computer equipped with an Intel Core i5 11th generation processor and 16,GB of RAM, with the implementation developed in Python 3. This study replicates the protocol of Badilla-Salamanca et al. (2025), which in turn is *inspired by* Valero-Carreras et al. (2023). We evaluate the algorithm on five benchmark instances, running each instance under two wall-clock limits (1200,s and 3600,s). For model assessment we employ 5-fold cross-validation (Kohavi, 1995): the dataset is partitioned into five equal folds; in each iteration one fold is held out for testing and the remaining four are used for training, cycling through all folds so that each serves exactly once as test data.

For every fold, we execute three independent runs with different random seeds. Each run yields a set of non-dominated solutions; per metric, we keep the best individual on the (minimum) Pareto front. In total, the design comprises 5 instances $\times$ 5 folds $\times$ 3 seeds $=$ 75 runs per algorithm.

In this section, we present the results obtained by NSGA-III and MOPSO, and compare them with the results previously reported by Badilla-Salamanca et al. (2025).

According to the experimental design, each experiment is executed 15 times (each of the 5 folds is run three times). For every fold, the best value for each metric is selected; therefore, each dataset yields five values per metric, which are subsequently averaged to obtain a representative performance score.

Table 3 reports, for each evaluation metric, the mean of the five maximum values obtained by each of the three algorithms under comparison. The first section of the table begins with two columns describing the dataset's rows and columns, respectively. This is followed by the AUC-ROC, F-Score, and CKC metrics for each algorithm, calculated based on the averaged maxima.

The second section of the table provides a summary of "wins" for each algorithm, indicating how many times it outperformed the other two across all datasets and metrics.

Table 3: Comparison of algorithms on large-scale datasets (1h).

Dataset	NSGA-II			NSGA-III			MOPSO		
	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC	AUC-ROC	F-Score	CKC
Software	0.751	0.874	0.018	0.773	**0.885**	0.386	**0.773**	0.884	**0.392**
Santander	0.646	0.227	0.072	0.661	0.250	0.145	**0.683**	**0.267**	**0.165**
Fraud	0.981	0.009	0.006	**0.982**	0.006	0.003	0.970	**0.831**	**0.831**
Higgs	0.566	0.434	0.094	**0.577**	**0.493**	0.089	0.574	0.483	**0.122**
SUSY	**0.832**	**0.731**	**0.527**	0.805	0.708	0.435	0.812	0.722	0.486
Hits against NSGA-II	-	-	-	4	4	2	3	4	4
Hits against NSGA-III	1	1	3	-	-	-	3	3	5
Hits against MOPSO	2	1	1	2	2	0	-	-	-
Total hits	**3**	**2**	**4**	**6**	**6**	**2**	**6**	**7**	**9**

To establish a clear comparison framework, we begin by examining the performance of NSGA-III relative to NSGA-II. The results show that NSGA-III achieves consistently higher values in both AUC-ROC and F-Score, outperforming NSGA-II in four out of the five evaluated datasets. This advantage is largely due to NSGA-III's reference-point based selection mechanism, which enhances diversity across the Pareto front and promotes solutions with superior class separability and more favorable precision–recall trade-offs. However, this superiority does not extend to Cohen's Kappa (CKC). In fact, NSGA-II attains higher CKC values in most cases. A plausible explanation is that NSGA-III often shifts its solutions toward the majority class to maximize F-Score, thereby increasing the expected agreement by chance—a component that CKC penalizes heavily. NSGA-II, on the other hand, tends to preserve a more balanced distribution of predictions across classes, resulting in a lower expected agreement and consequently higher Kappa values, even when its AUC or F1 performance is comparatively lower.

When extending the comparison to MOPSO versus NSGA-III, MOPSO demonstrates competitive performance in AUC-ROC and F-Score while clearly dominating in CKC across all datasets. This behavior is consistent with the algorithmic properties of particle swarm optimization. MOPSO benefits from a substantially lower computational overhead and a faster update mechanism, which enables broader exploration of the solution space within fewer iterations. Unlike NSGA-II and NSGA-III, MOPSO does not rely on computationally expensive non-dominated sorting procedures. As a result, it is less prone to converge toward solutions that overfit the majority class. The swarm's exploratory dynamics facilitate the discovery of more balanced decision structures, ultimately improving CKC. In summary, while NSGA-III tends to excel in discrimination-focused metrics such as AUC and F1, MOPSO achieves a more holistic performance by generating models with a better balance between predicted and true class distributions, leading to superior agreement as measured by Cohen's Kappa.

Notably, the Fraud dataset exhibits a particular behavior that highlights the structural differences between the evaluation metrics used. AUC-ROC, by averaging classifier performance across all possible decision thresholds and both classes equally, produces a dilution effect in the presence of class imbalance: a classifier that dominates the majority class can achieve a high value on this metric even when its ability to recognize the minority class is practically null. F-Score and CKC, on the other hand, are inherently sensitive to this imbalance. F-Score directly penalizes errors on the minority class, while CKC measures chance-corrected agreement, which drops sharply when predictions are biased toward the dominant class. This phenomenon explains the divergence observed in heavily imbalanced datasets such as Fraud.

6.4. Statistical Significance Analysis

To rigorously evaluate the performance differences among the three algorithms (NSGA-II, NSGA-III, and MOPSO) across the five datasets, a statistical significance analysis was conducted following the methodologies recommended for comparing classifiers over multiple datasets. Given that the results of metaheuristic algorithms do not necessarily follow normal distributions and that multiple algorithms are compared across multiple datasets, non-parametric tests based on rankings were employed, following the validation structure of Badilla-Salamanca et al. (2025).

The Friedman test (Friedman, 1940) was applied to detect significant differences among the algorithms considering three evaluation metrics: AUC-ROC, F-Score, and Cohen's Kappa Coefficient. Additionally, the Iman–Davenport correction (Iman & Davenport, 1980) was applied. This adjustment transforms the Friedman statistic into an F-statistic, leading to more reliable results, particularly in scenarios involving a limited number of algorithms or datasets.

When the Iman-Davenport test indicated significant differences ($p < 0.05$), pairwise comparisons were conducted using the Wilcoxon signed-rank test Wilcoxon (1945), applying the Bonferroni correction to adjust the significance level and control Type I error in multiple comparisons. This approach allows identification of which pairs of algorithms exhibit statistically significant differences.

In addition to statistical significance, effect size was calculated using the Vargha-Delaney A_{12} statistic Vargha & Delaney (2000). This non-parametric measure quantifies the probability that one algorithm outperforms another in terms of performance, providing a practical interpretation of the magnitude of observed differences. Values of $A_{12} > 0.5$ indicate that the first algorithm tends to outperform the second, while values < 0.5 suggest that the second algorithm outperforms the first; values close to 0.5 indicate similar performance between the two algorithms.

The results of the statistical significance analysis for each metric are presented in Tables 4, 5, and 6. For each metric, the following are reported: (i) p-values from the Friedman and Iman-Davenport tests, (ii) p-values from pairwise Wilcoxon comparisons, and (iii) A_{12} effect size values between each pair of algorithms.

6.5. Analysis of Results

The Friedman and Iman-Davenport tests revealed statistically significant differences among the three algorithms for all evaluated metrics ($p < 0.001$). The pairwise Wilcoxon comparisons, combined with the A_{12} effect sizes, provide deeper insights into the relative performance of each algorithm pair.

In Table 4 is the result for the AUC-ROC metric, both NSGA-III and MOPSO significantly outperformed NSGA-II ($p < 0.001$), with A_{12} values of 0.413 and 0.415 respectively, indicating that NSGA-II achieves superior AUC-ROC values in approximately 58% of comparisons. No significant difference was observed between NSGA-III and MOPSO ($p = 0.198$).

Table 4: Statistical results for the AUC metric

Global test	p**-value**	
Friedman	2.32×10^{-6}	
Iman–Davenport	7.88×10^{-7}	
Comparison	**Wilcoxon** p**-value**	A_{12}
NSGA-II vs NSGA-III	2.29×10^{-5}	0.413
NSGA-II vs MOPSO	4.20×10^{-4}	0.415
NSGA-III vs MOPSO	0.1976	0.488

Regarding the F1-Score presented in Table 5, all pairwise comparisons showed significant differences. The A_{12} values indicate that NSGA-III substantially outperforms NSGA-II ($A_{12} = 0.184$) and MOPSO ($A_{12} = 0.361$), while MOPSO also significantly outperforms NSGA-II ($A_{12} = 0.067$).

Table 5: Statistical results for the F1 metric

Global test	p**-value**	
Friedman	1.65×10^{-24}	
Iman–Davenport	1.11×10^{-16}	
Comparison	**Wilcoxon** p**-value**	A_{12}
NSGA-II vs NSGA-III	8.91×10^{-14}	0.184
NSGA-II vs MOPSO	5.28×10^{-14}	0.067
NSGA-III vs MOPSO	2.69×10^{-4}	0.361

For Cohen's Kappa Coefficient (Table 6), NSGA-II demonstrated superior performance compared to NSGA-III ($A_{12} = 0.723$, $p < 0.001$) and marginally better than MOPSO ($A_{12} = 0.557$, $p = 0.055$). Additionally, MOPSO significantly outperformed NSGA-III ($p < 0.001$, $A_{12} = 0.252$).

Table 6: Statistical results for the CKC metric

Global test	p**-value**	
Friedman	4.39×10^{-20}	
Iman–Davenport	1.11×10^{-16}	
Comparison	**Wilcoxon** p**-value**	A_{12}
NSGA-II vs NSGA-III	1.60×10^{-12}	0.723
NSGA-II vs MOPSO	0.0553	0.557
NSGA-III vs MOPSO	6.05×10^{-11}	0.252

These results demonstrate that no single algorithm dominates across all metrics, highlighting the multi-objective nature of the problem and the importance of considering multiple performance indicators when evaluating classification algorithms.

7. Conclusions

This research addressed the problem of feature selection for FS-SVM from a metaheuristic perspective. The NSGA-III and MOPSO algorithms were introduced and examined in detail, including their operators, internal mechanisms, and the way in which each of them formulates and solves a multi-objective problem involving five simultaneous optimization functions. This formulation enabled the modeling of different dimensions of classifier performance, as well as the cost and structural complexity associated with the selected feature subsets.

Following the methodological framework established in Badilla-Salamanca et al. (2025), a direct comparison was conducted between the proposed algorithms and those reported in the existing literature, ensuring experimental consistency and statistical robustness. The results demonstrated clear improvements over the performance achieved previously by NSGA-II. Furthermore, it was found that explicitly incorporating the performance metrics as optimization objectives significantly enhances algorithmic performance, as it aligns the search process with the actual classification quality and reduces the dependence on post-hoc threshold adjustments.

Among the evaluated methods, MOPSO exhibited superior performance in most scenarios, largely due to its lower computational overhead and its swarm-based optimization paradigm, which supports broader and more efficient exploration of the search space. This exploratory advantage proved particularly beneficial in high-dimensional and imbalanced datasets.

Overall, the findings of this study contribute to a deeper understanding of how different evolutionary and population-based strategies behave in complex, multi-objective classification problems. Several directions for future research emerge from this work. First, the exploration of additional population-based metaheuristics beyond genetic and swarm-based paradigms — such as differential evolution or ant colony optimization — could provide further insights into the trade-offs between computational efficiency and solution quality in the SVM-FS context. Second, a systematic parameter calibration study for both NSGA-III and MOPSO could yield more refined algorithm configurations, potentially enhancing their performance across diverse dataset characteristics. Third, extending the proposed framework toward a more comprehensive data preprocessing pipeline represents a promising avenue; rather than solely selecting features, future algorithms could simultaneously optimize preprocessing decisions such as normalization strategies, missing value imputation, and outlier treatment, thereby addressing the broader challenge of automated machine learning in large-scale settings. Finally, the scalability of the proposed approaches could be further investigated in the context of big data environments, where distributed computing frameworks and online learning strategies may be required to handle datasets of increasing volume and dimensionality.

References

Abasabadi, S., Nematzadeh, H., Motameni, H., & Akbari, E. (2022). Hybrid feature selection based on sli and genetic algorithm for microarray datasets. *The Journal of Supercomputing*, *78*, 19725–19753. doi:10.1007/s11227-022-04650-w.

Afreen, S., Bhurjee, A. K., & Aziz, R. M. (2025). Feature selection using game shapley improved grey wolf optimizer for optimizing cancer classification. *Knowledge and Information Systems*, (pp. 1–32). doi:10.1007/s10115-025-02340-6.

Alcaraz, J. (2024). Redesigning a NSGA-II metaheuristic for the bi-objective support vector machine with feature selection. *Computers & Operations Research*, *172*, 106821. doi:10.1016/j.cor.2024.106821.

Alcaraz, J., Labbé, M., & Landete, M. (2022). Support vector machine with feature selection: A multiobjective approach. *Expert Systems with Applications*, *204*, 117485. doi:10.1016/j.eswa.2022.117485.

Awoyemi, J. O., Adetunmbi, A. O., & Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)* (pp. 1–9). doi:10.1109/ICCNI.2017.8123782.

Aytug, H. (2015). Feature selection for support vector machines using generalized benders decomposition. *European Journal of Operational Research*, *244*, 210–218. doi:10.1016/j.ejor.2015.01.006.

Azhari, M., Abarda, A., Ettaki, B., Zerouaoui, J., & Dakkon, M. (2020). Higgs boson discovery using machine learning methods with pyspark. *Procedia Computer Science*, *170*, 1141–1146. doi:10.1016/j.procs.2020.03.053.

Badilla-Salamanca, M., Medina Durán, R., & Contreras-Bolton, C. (2025). An effective multi-objective metaheuristic for the support vector machine with feature selection. *Knowledge-Based Systems*, *328*, 114203. doi:https://doi.org/10.1016/j.knosys.2025.114203.

Baldi, P., Sadowski, P., & Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, *5*, 4308. doi:10.1038/ncomms5308.

Baldomero-Naranjo, M., Martínez-Merino, L. I., & Rodríguez-Chía, A. M. (2021). A robust SVM-based approach with feature selection and outliers detection for classification problems. *Expert Systems with Applications*, *178*, 115017. doi:10.1016/j.eswa.2021.115017.

Benítez-Peña, S., Blanquero, R., Carrizosa, E., & Ramírez-Cobo, P. (2019). Cost-sensitive feature selection for support vector machines. *Computers & Operations Research*, *106*, 169–178. doi:10.1016/j.cor.2018.03.005.

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. (1st ed.). Berlin, Heidelberg: Springer.

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

Bote-Curiel, L., Ruiz-Llorente, S., Muñoz-Romero, S., Yagüe-Fernández, M., Barquín, A., García-Donas, J., & Rojo-Álvarez, J. L. (2022). Multivariate feature selection and autoencoder embeddings of ovarian cancer clinical and genetic data. *Expert Systems with Applications*, *206*, 117865. doi:10.1016/j.eswa.2022.117865.

Bouchlaghem, Y., Akhiat, Y., Touchanti, K., & Amjad, S. (2024). A novel feature selection method with transition similarity measure using reinforcement learning. *Decision Analytics Journal*, *11*, 100477. doi:10.1016/j.dajour.2024.100477.

Bouraoui, A., Jamoussi, S., & BenAyed, Y. (2018). A multi-objective genetic algorithm for simultaneous model and feature selection for support vector machines. *Artificial Intelligence Review*, *50*, 261–281. doi:10.1007/s10462-017-9543-9.

Candelieri, A., Giordani, I., Archetti, F., Barkalov, K., Meyerov, I., Polovinkin, A., Sysoyev, A., & Zolotykh, N. (2019). Tuning hyperparameters of a SVM-based water demand forecasting system through parallel global optimization. *Computers & Operations Research*, *106*, 202–209. doi:10.1016/j.cor.2018.01.013.

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

Coello Coello, C., & Lechuga, M. (2002). Mopso: a proposal for multiple objective particle swarm optimization. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)* (pp. 1051–1056 vol.2). volume 2. doi:10.1109/CEC.2002.1004388.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, *20*, 37–46. doi:10.1177/001316446002000104.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*, 273–297. doi:10.1023/A:1022627411411.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*, 21–27. doi:10.1109/TIT.1967.1053964.

Deb, K., & Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, *18*, 577–601. doi:10.1109/TEVC.2013.2281535.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*, 182–197. doi:10.1109/4235.996017.

Dudzik, W., Nalepa, J., & Kawulok, M. (2021). Evolving data-adaptive support vector machines for binary classification. *Knowledge-Based Systems*, *227*, 107221. doi:10.1016/j.knosys.2021.107221.

Faris, H., Hassonah, M. A., Al-Zoubi, A. M., Mirjalili, S., & Aljarah, I. (2018). A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a robust system architecture. *Neural Computing and Applications*, *30*, 2355–2369. doi:10.1007/s00521-016-2818-2.

Farokhmanesh, F., & Sadeghi, M. T. (2021). Deep neural networks regularization using a combination of sparsity inducing feature selection methods. *Neural Processing Letters*, *53*, 701–720. doi:10.1007/s11063-020-10389-3.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, *11*, 86–92.

Gaudioso, M., Gorgone, E., Labbé, M., & Rodríguez-Chía, A. (2017). Lagrangian relaxation for SVM feature selection. *Computers & Operations Research*, *87*, 137–145. doi:10.1016/j.cor.2017.06.001.

Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A. (2006). *Feature Extraction: Foundations and Applications*. Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-540-35488-8.

Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, *143*, 29–36. doi:10.1148/radiology.143.1.7063747.

Huang, T., Ngan, C.-K., Cheung, Y. T., Marcotte, M., & Cabrera, B. (2025). 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 Bioinformatics and Biotechnology*, *6*. doi:10.2196/65001.

Iman, R. L., & Davenport, J. M. (1980). Approximations of the critical region of the fbietkan statistic. *Communications in Statistics - Theory and Methods*, *9*, 571–595. doi:10.1080/03610928008827904. arXiv:https://doi.org/10.1080/03610928008827904.

Jain, A., & Jain, V. (2022). Sentiment classification using hybrid feature selection and ensemble classifier. *Journal of Intelligent & Fuzzy Systems*, *42*, 659–668. doi:10.3233/JIFS-189738.

Junaid, H. H. S., Daneshfar, F., & Mohammad, M. A. (2025). Automatic colorectal cancer detection using machine learning and deep learning based on feature selection in histopathological images. *Biomedical Signal Processing and Control*, *107*, 107866. doi:10.1016/j.bspc.2025.107866.

Kaggle (2024). Kaggle: High-quality public datasets. URL: https://www.kaggle.com/ accessed: 2024-12-20.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks* (pp. 1942–1948 vol.4). volume 4. doi:10.1109/ICNN.1995.488968.

Kim, H., & Kim, K.-H. (2024). Deep learning-based sbom defect detection for medical devices. In *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* (pp. 047–051). doi:10.1109/ICAIIC60209.2024.10463483.

Kohavi, R. (1995). 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 (pp. 1137–1143). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Labbé, M., Martínez-Merino, L. I., & Rodríguez-Chía, A. M. (2019). Mixed integer linear programming for feature selection in support vector machine. *Discrete Applied Mathematics*, *261*, 276–304. doi:10.1016/j.dam.2018.10.025.

Lee, I. G., Zhang, Q., Yoon, S. W., & Won, D. (2020). A mixed integer linear programming support vector machine for cost-effective feature selection. *Knowledge-Based Systems*, *203*, 106145. doi:10.1016/j.knosys.2020.106145.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys*, *50*. doi:10.1145/3136625.

Maldonado, S., Pérez, J., Weber, R., & Labbé, M. (2014). Feature selection for support vector machines via mixed integer linear programming. *Information Sciences*, *279*, 163–175. doi:10.1016/j.ins.2014.03.110.

Mavrotas, G., & Florios, K. (2013). An improved version of the augmented ϵ-constraint method (augmecon2) for finding the exact pareto set in multi-objective integer programming problems. *Applied Mathematics and Computation*, *219*, 9652–9669. doi:10.1016/j.amc.2013.03.002.

Miao, J., & Niu, L. (2016). A survey on feature selection. *Procedia Computer Science*, *91*, 919–926. doi:10.1016/j.procs.2016.07.111. Promoting Business Analytics and Quantitative Management of Technology: 4th International Conference on Information Technology and Quantitative Management (ITQM 2016).

Mohammed, R., Rawashdeh, J., & Abdullah, M. (2020). Machine learning with oversampling and undersampling techniques: Overview study and experimental results. In *2020 11th International Conference on Information and Communication Systems (ICICS)* (pp. 243–248). doi:10.1109/ICICS49469.2020.239556.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81–106. doi:10.1007/BF00116251.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*, 533–536. doi:10.1038/323533a0.

Sabzekar, M., & Aydin, Z. (2021). A noise-aware feature selection approach for classification. *Soft Computing*, *25*, 6391–6400. doi:10.1007/s00500-021-05630-7.

Sabzekar, M., Yazdi, H. S., & Naghibzadeh, M. (2012). Relaxed constraints support vector machine. *Expert Systems*, *29*, 506–525. doi:10.1111/j.1468-0394.2011.00611.x.

Valero-Carreras, D., Alcaraz, J., & Landete, M. (2023). Comparing two svm models through different metrics based on the confusion matrix. *Computers and Operations Research*, *152*, 106131. doi:10.1016/j.cor.2022.106131.

Van Rijsbergen, C. J. (1979). *Information Retrieval*. (2nd ed.). USA: Butterworth-Heinemann.

Vapnik, V., & Chervonenkis, A. (1964). A note on one class of perceptions. *Automation and Remote Control*, *25*. The original article is in Russian.

Vapnik, V., & Chervonenkis, A. (1974). *Theory of Pattern Recognition*. Moscow: Nauka. The original article is in Russian.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Berlin, Heidelberg: Springer-Verlag.

Vargha, A., & Delaney, H. D. (2000). A critique and improvement of the cl common language effect size statistics of mcgraw and wong. *Journal of Educational and Behavioral Statistics*, *25*, 101–132. doi:10.3102/10769986025002101.

Wang, J., Wang, X., Li, X., & Yi, J. (2023). 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*. doi:10.3390/e25030531.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*, 80–83. doi:10.2307/3001968.

Xue, B., Zhang, M., Browne, W. N., & Yao, X. (2016). A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, *20*, 606–626. doi:10.1109/TEVC.2015.2504420.

Xue, Y., & Zhang, C. (2024). A novel importance-guided particle swarm optimization based on mlp for solving large-scale feature selection problems. *Swarm and Evolutionary Computation*, *91*, 101760. doi:10.1016/j.swevo.2024.101760.

Xue, Y., Zhang, C., Neri, F., Gabbouj, M., & Zhang, Y. (2023). An external attention-based feature ranker for large-scale feature selection. *Knowledge-Based Systems*, *281*, 111084. doi:10.1016/j.knosys.2023.111084.

Zandvakili, A., Javidi, M. M., & Mansouri, N. (2024). Simultaneous feature selection and SVM optimization based on fuzzy signature and chaos goa. *Evolving Systems*, *15*, 1907–1937. doi:10.1007/s12530-024-09595-4.